Solving Partial Differential Equations Using Artificial Neural Networks

by

Keith Rudd

Department of Mechanical Engineering and Material Sciences
Duke University

Date: _____

Approved:

_____
Silvia Ferrari, Supervisor

_____
Edward J. Shaughnessy

_____
John D. Albertson

_____
Xiaobai Sun

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Mechanical Engineering and Material Sciences
in the Graduate School of Duke University
2013

Abstract

Solving Partial Differential Equations Using Artificial Neural Networks

by

Keith Rudd

Department of Mechanical Engineering and Material Sciences
Duke University

Date: _____

Approved:

_____
Silvia Ferrari, Supervisor

_____
Edward J. Shaughnessy

_____
John D. Albertson

_____
Xiaobai Sun

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Mechanical Engineering and Material Sciences
in the Graduate School of Duke University
2013

Abstract

This thesis presents a method for solving partial differential equations (PDEs) using artificial neural networks. The method uses a constrained-backpropagation (CPROP) approach for preserving prior knowledge during incremental training for solving nonlinear elliptic and parabolic PDEs adaptively, in non-stationary environments. Compared to previous methods that use penalty functions or Lagrange multipliers, CPROP reduces the dimensionality of the optimization problem by using direct elimination, while satisfying the equality constraints associated with the boundary and initial conditions exactly, at every iteration of the algorithm. The effectiveness of this method is demonstrated through several examples, including nonlinear elliptic and parabolic PDEs with changing parameters and non-homogeneous terms. The computational complexity analysis shows that CPROP compares favorably to existing methods of solution, and that it leads to considerable computational savings when subject to non-stationary environments.

The CPROP based approach is extended to a constrained integration (CINT) method for solving initial boundary value partial differential equations (PDEs). The CINT method combines classical Galerkin methods with CPROP in order to constrain the ANN to approximately satisfy the boundary condition at each stage of integration. The advantage of the CINT method is that it is readily applicable to PDEs in irregular domains and requires no special modification for domains with complex geometries. Furthermore, the CINT method provides a semi-analytical so-

lution that is infinitely differentiable. The CINT method is demonstrated on two hyperbolic and one parabolic initial boundary value problems (IBVPs). These IBVPs are widely used and have known analytical solutions. When compared with Matlab's finite element (FE) method, the CINT method is shown to achieve significant improvements both in terms of computational time and accuracy.

The CINT method is applied to a distributed optimal control (DOC) problem of computing optimal state and control trajectories for a multiscale dynamical system comprised of many interacting dynamical systems, or agents. A generalized reduced gradient (GRG) approach is presented in which the agent dynamics are described by a small system of stochastic differential equations (SDEs). A set of optimality conditions is derived using calculus of variations, and used to compute the optimal macroscopic state and microscopic control laws. An indirect GRG approach is used to solve the optimality conditions numerically for large systems of agents. By assuming a parametric control law obtained from the superposition of linear basis functions, the agent control laws can be determined via set-point regulation, such that the macroscopic behavior of the agents is optimized over time, based on multiple, interactive navigation objectives.

Lastly, the CINT method is used to identify optimal root profiles in water limited ecosystems. Knowledge of root depths and distributions is vital in order to accurately model and predict hydrological ecosystem dynamics. Therefore, there is interest in accurately predicting distributions for various vegetation types, soils, and climates. Numerical experiments were were performed that identify root profiles that maximize transpiration over a 10 year period across a transect of the Kalahari. Storm types were varied to show the dependence of the optimal profile on storm frequency and intensity. It is shown that more deeply distributed roots are optimal for regions where storms are more intense and less frequent, and shallower roots are advantageous in regions where storms are less intense and more frequent.

I dedicate this thesis to my loving wife and son, who have been a great source of strength and motivation.

Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

Symbols

$\mathcal{B}$	Linear differential operator in the boundary condition.
$\mathbf{B}_L$	Matrix resulting from applying $\mathcal{B}$ to LTM transfer functions and evaluating at points in $\mathcal{T}_L$.
$\mathbf{B}_S$	Matrix resulting from applying $\mathcal{B}$ to STM transfer functions and evaluated at points in $\mathcal{T}_L$.
$\mathbf{b}$	Neural network input bias.
$\mathcal{C}$	Function that explicitly gives the long term memory weights as a function of short term memory weights.
$\mathcal{D}$	Differential operator.
e	Error or objective function of long term and short term weights.
E	Error or objective function of short term memory weights.
$\mathcal{E}$	Evaporation rate.
F	Forcing function or non-homogeneity associated with $\mathcal{D}$ in boundary value problems.
f	Function used to specify the boundary condition.
$\tilde{f}$	Twice differentiable function equal to f along the domain boundary.
$\mathbf{g}$	Constraints on the network weights used during incremental training.
G	The function resulting from applying the differential operator to the approximate solution.
$\mathcal{I}$	A subset of the real numbers.

$\mathbf{I}$	Identity matrix.
h	Initial condition(s).
$\mathbf{h}$	Vector containing the initial condition evaluated at points along the boundary.
$\mathbf{J}$	Jacobian matrix of vector $\boldsymbol{\epsilon}$.
J	Cost function.
K	Hydraulic conductivity.
$\mathbf{M}$	Matrix resulting from evaluating the approximate solution in the constrained integration method at collocation points within the domain.
N	Number of training points.
N_L	Number of LTM training points.
N_S	Number of STM training points.
$\mathbf{p}_\ell$	Initial condition(s) arising in the system of ordinary differential equations in the
$\wp$	Probability density function of agents' locations.
PET	Potential evapotranspiration.
PE	Potential evaporation.
PT	Potential transpiration.
Q	Number of nodes in the hidden layer of the artificial neural network approximate partial differential equation solution.
Q_L	Number of long term memory nodes in the hidden layer of the artificial neural network approximate partial differential equation solution.
Q_S	Number of short term memory nodes in the hidden layer of the artificial neural network approximate partial differential equation solution.
q	Function that is zero along the domain boundary and non-zero over the domain interior.
$\mathcal{R}$	Rainfall rate.

S	Root sink term.
$\mathcal{T}$	Training set.
T	Total transpiration.
$\mathcal{T}_L$	Long term memory training set.
$\mathcal{T}_S$	Short term memory training set.
t	Time.
u	PDE solution.
$\hat{u}$	Approximate (numerical) PDE solution.
$\mathbf{u}$	Control vector.
$\mathbf{v}$	Vector of network output weights.
$\mathbf{v}_L$	Vector of network long term memory output weights.
$\mathbf{v}_S$	Vector of network short term memory output weights.
$\mathbf{w}$	Vector of network weights.
$\mathbf{w}_L$	Vector of network long term memory weights.
$\mathbf{w}_S$	Vector of network short term memory weights.
$\mathbf{W}$	Matrix of network input weights.
$\mathbf{W}_L$	Matrix of network long term memory input weights.
$\mathbf{W}_S$	Matrix of network short term memory input weights.
$\mathbf{x}_k$	Input training point in $\mathcal{T}$.
$\mathbf{x}_\ell$	Input training point in $\mathcal{T}_L$.
$\mathbf{x}_S$	Input training point in $\mathcal{T}_S$.
$\mathbf{y}_k$	Output training point.
$\hat{y}$	Neural network output.
z	Depth, with positive upwards.
α^*	Mean storm depth.
$\boldsymbol{\epsilon}$	Vector of errors corresponding to points in $\mathcal{T}_S$.

η	Learning rate.
χ	A partial derivative of u.
$\hat{\chi}$	The neural network approximation of χ.
σ	Neural network transfer function.
ξ_1	A term in the derivative of the function G with respect to the network weights.
ξ_2	A term in the derivative of the function G with respect to the network weights.
$\boldsymbol{\xi}$	Vector containing the evaluation of $\mathcal{D}(\hat{u})$ at points in $\mathcal{T}_S$.
$\mathbf{\Lambda}_S$	A product of diagonal matrices taken from the columns of $\mathbf{W}_S$.
$\mathbf{\Lambda}_L$	A product of diagonal matrices taken from the columns of $\mathbf{W}_L$.
λ_3	Third zero of the third order Bessel function of the first kind.
λ_4	Fourth zero of the zeroth order Bessel function of the first kind.
λ^*	Mean storm frequency.
ψ	Hydraulic pressure head.
θ	Volumetric water content.
ρ	Root density.
γ	Root efficiency term.
ζ	The term found in the ANN approximate solution in the CINT method that is independent of $\mathbf{v}_S$.
LAI	Leaf area index.

Abbreviations

ANN	Artificial neural network.
BC	Boundary condition.
BVP	Boundary value problem.
CINT	Constrained integration.

CPROP	Constrained backpropagation.
DOC	Distributed optimal control.
FDM	Finite difference method.
FEM	Finite element method.
GRG	Generalized reduced gradient.
HJB	Hamilton-Jacobi-Bellman equation.
IBVP	Initial boundary value problem.
IC	Initial condition.
LM	Levenberg-Marquardt.
LTM	Long term memory from prior training of an ANN.
ODE	Ordinary differential equation.
PDE	Partial differential equation.
VLRS	Very large robotic systems
SDE	Stochastic differential equation.
STM	Short term memory to be learned by the ANN.

Acknowledgements

I would first like to thank my adviser, Dr. Silvia Ferrari, for your support over the course of my time at Duke. Without your guidance, financial support, and encouragement this thesis would never have come to fruition.

I would like to thank the members of my committee for their time and willingness to aid me in my academic pursuits. Specifically, I would like to thank Dr. John Albertson for his invaluable help in understanding hydrology, and Dr. Xiaobai Sun and Dr. Edward Shaughnessy for your encouragement and input in directing my research. In addition, I would like to thank Dr. Wilkins Aquino for helping me understanding inverse problems.

I am grateful for the financial support I have received from the NSF through the IGERT WiseNet program, and to Amy Yonai for all her work in that program. Lastly, I would like to thank my lab mates that have provided very valuable feedback and support: Greg Foderaro, Ashleigh Swingler, Wenjie Lu, Xu Zhang, Hongchuan Wei, and Weston Ross.

1

Introduction

Effective methods, such as the finite difference method (FDM) and the finite element method (FEM), have been developed for solving partial differential equation (PDE) problems numerically in stationary environments [102, 42]. Given a PDE with known parameters and initial and boundary conditions, FDM and FEM algorithms compute the approximate value of the solution at a discrete number of points, producing a look-up table that can be interpolated when the solution is needed elsewhere in the domain. One disadvantage of these methods is that, in order to obtain satisfactory solution accuracy, it may be necessary to deal with fine meshes that significantly increase the size of the look-up table and memory required [80, 88].

ANNs provide an ideal representation tool for adaptive PDE solutions because they are characterized by adjustable parameters that can be modified by incremental training algorithms [1], and because of their ability to approximate nonlinear functions on a compact space. Furthermore, ANN solutions of PDEs are characterized by other advantages over FDM and FEM solutions that are especially important in non-stationary environments. One advantage is that the approximate solution is given in closed analytic form and is infinitely differentiable. This solution is represented by a

small number of parameters, which reduces the amount of memory required [56, 97]. Another advantage is that the solution is valid over the entire domain, eliminating the need for interpolation.

Artificial neural networks (ANNs) are commonly implemented to provide a functional representation of PDE solutions. Examples range from solving the Poisson equation [55] to solving the Hamilton-Jacobi-Bellman (HJB) equation to find the fixed-final-time-constrained optimal control for nonlinear systems [20]. In many of these applications, the PDE describes an underlying dynamic process that is subject to change as a result of a non-stationary environment. Therefore, while the PDE may capture the dynamic process on short time scales, the process, and thus the PDE, both are subject to change over long time scales. In particular, for a given dynamic process, a non-stationary environment may bring about incremental changes in the PDE parameters, and external forcing (or nonhomogeneous term) that result in incremental changes of the PDE solution. An adaptive PDE solution can respond to these changes by adapting incrementally over time to satisfy the PDE problem subject to changing parameters, and/or changing external forcing.

One approach to solving PDEs numerically using ANNs is to utilize the discrete FDM or FEM solution to train a neural network using a conventional backpropagation algorithm [59], such as Levenberg-Marquardt (LM), in batch mode [62]. Methods have also been proposed to determine the PDE solution in one step by training an ANN to minimize an error function formulated in terms of the differential operator. One of the main difficulties that arises in ANN-based methods lies in satisfying the boundary conditions (BCs) and initial conditions (ICs), which amount to a set of equality constraints on a continuous domain. One possibility is to use a problem-specific ansatz that has been tailored to automatically satisfy BCs, while containing an ANN that is trained to minimize the PDE error. Although this approach has been shown effective at solving boundary value problems (BVPs) with a high degree of ac-

curacy [55, 95, 70], it has yet to be demonstrated on initial-boundary value problems (IBVPs). Another disadvantage is that the ansatz is problem specific and, thus, it has to be designed by the user. As a result, the approach may not be applicable to all PDE problems, and cannot be used to obtain an adaptive PDE solution.

Another approach for incorporating the ICs and BCs in the ANN solution is to use them to formulate a penalty function, thereby converting the constrained optimization problem into an unconstrained optimization problem [27]. As with all penalty function methods [33], this method can display slow convergence, and poor solution accuracy in the equality constraints (I/BCs). Improving accuracy typically requires using many more nodes in the ANN hidden layer, and a dense set of collocation points along the boundary of the domain. Besides making the approach computationally expensive, these steps involve user intervention, and, therefore, do not allow for an adaptive solution of the PDE problem.

A well known result from constrained optimization theory is that if the equality constraints satisfy the implicit function theorem, they can be at once satisfied exactly, and used to reduce the dimensionality of the optimization problem, through the method of direct elimination [103, 8]. Thus, whenever applicable, direct elimination is to be preferred over the penalty function method or the method of Lagrange multipliers, which rely on augmenting the objective function by a function of the constraints and, thus, increase the dimensionality of the unconstrained optimization by introducing additional variables (e.g. Lagrange multipliers).

It was recently shown in [31] that the method of direct elimination can be used to train ANNs in the presence of equality constraints through a method known as constrained backpropagation (CPROP). CPROP preserves a set of input-output and gradient information during incremental training sessions by embedding this information into a set of equality constraints that are formulated in terms of the neural weights by means of algebraic training [32]. In previous work, CPROP has been

used to eliminate interference [12], and preserve prior knowledge in fully-connected sigmoidal neural networks, and to adapt an ANN-based nonlinear controller online, subject to changing and unmodeled aircraft dynamics [31]. In [26], CPROP was demonstrated on benchmark problems in function approximation, system identification, and the solution of ordinary differential equations (ODEs).

This thesis shows that CPROP offers a natural paradigm for solving PDEs via ANNs, because the ANN can be adapted to minimize the error defined by the differential operator, while satisfying the I/BC equality constraints. Furthermore, since it allows for the equality constraints to be satisfied during incremental training sessions, CPROP can be easily extended to the adaptive solution of PDEs in non-stationary environments. In the case of elliptic BVPs, CPROP equality constraints are used to preserve the BC. As a result, the shape of the domain does not increase the difficulty of the method, and the CPROP algorithm is more computationally efficient and converges more rapidly than other ANN based methods, because there is no need for a penalty function. In the case of parabolic IBVPs, the solution is structured so as to satisfy the BC exactly, while the IC is preserved using the equality constraints. It is shown that adaptive CPROP solutions brings about a significant reduction in computation time compared to existing methods [44] for elliptic and parabolic PDEs.

To further improve speed and accuracy in obtaining numerical solutions for stationary PDE problems, the CPROP PDE adaptive solution method is extended to a constrained integration (CINT) method, in which CPROP is combined with traditional Galerkin methods. Galerkin methods have been used in ANN training in order to overcome slow convergence. For example, in [71] the ANN output weights were found using an inner product rather than a more traditional training method, such as backpropagation or genetic algorithm. A similar approach was used in [20] to solve the HJB equation, and in [48] to analyze bifurcations of a cellular nonlinear network. In these cases, the ANN output weights were treated as functions of time and an

inner product was used to transform the PDEs into systems of ODEs. The output weights were then found by integrating the resulting ODEs. Using Galerkin methods for training has been shown to improve solution accuracy and computational time when compared to traditional training methods [71, 20, 48]. However, one disadvantage of continuous Galerkin methods is that PDEs solved over non-rectangular or irregular domains typically require a domain transformation [53] or domain decomposition [75], which greatly increases the difficulty of obtaining an approximate solution. A smoothed boundary method was proposed in [15] to overcome the difficulties associated with irregular domains for PDE problems with a zero-flux BC. In smoothed boundary methods the domain is embedded into a box and a smoothing term is used to encode the boundary condition into a modified PDE that can be solved using standard Galerkin methods.

The CINT method is broadly applicable to solving problems in irregular domains, eliminating the need to perform a domain transformation or decomposition, or modify the PDE as done in [15]. Additionally, the CINT method is applicable to problems with Dirichlet, Neumann, and/or Robin boundary conditions. Like in the inner product based methods used in [20, 48], the CINT method approximates the solution with a single layered ANN with time-dependent output weights. However, unlike the ANN based approach developed in [20, 48], which does not directly address how boundary conditions are satisfied, the CINT method utilizes CPROP to constrain the the ANN so as to satisfy the boundary condition.

This thesis is organized as follows. Chapter 2 defines the classes of PDE problems that are addressed in this thesis and gives a background of the CPROP and Galerkin methods. In Chapter 3, the CPROP and CINT methods for solving PDEs are given and demonstrated through several example problems in Chapter 4, including Laplace's equation, the heat/diffusion equation in two and three spatial dimensions, the Boussinesq equation, the wave equation. In Chapter 5 the CINT method is

applied to the problem of obtaining the optimal control of a multi-scale dynamical system comprised of many interacting agents, and in Chapter 6 the CINT method is used to identify the optimal root profile in water-limited ecosystems.

2

Problem Formulation and Background

PDE problems frequently arise in areas such as fluid mechanics, thermodynamics, and optimal control that can greatly benefit from ANN solution methods that approximate the PDE solution with an infinitely differentiable, close form solution. This thesis presents new CPROP and CINT methodologies that use ANN to numerically solve linear and nonlinear elliptic BVPs and parabolic and hyperbolic IBVPs of second order. Section 2.1 describes the classes of PDEs that to which the CPROP and CINT methods are applicable.

This chapter also includes background information on the CPROP method in Section 2.2. The CPROP method preserves a set of input-output and gradient information during incremental training sessions by embedding this information into a set of equality constraints that are formulated in terms of the neural weights by means of algebraic training [32]. It will be shown in Chapter 3, that CPROP is used to constrain the ANN such that the BC or IC is satisfied at each iteration of training.

Lastly, this chapter gives background information on Galerkin's method in Section 2.3. The CINT method is a modification of Galerkin's method and is used to solve parabolic and hyperbolic IBVPs. In the CINT method, the output ANN weights are

trained using an inner-product, as done in Galerkin's method, however, the output weights are constrained using a modified CPROP method, such that the ANN PDE solution satisfies the PDE's BC at each step of integration.

2.1 Problem Formulation

This dissertation gives new methods that use ANN to approximate solutions to linear and nonlinear second order PDEs of elliptic, parabolic, and hyperbolic type. A linear second order PDE over the domain $\Omega \subset \mathbb{R}^2$ has the form,

$$a(\xi,\eta)\frac{\partial^2 u}{\partial \xi^2} + b(\xi,\eta)\frac{\partial^2 u}{\partial \xi \partial \eta} + c(\xi,\eta)\frac{\partial^2 u}{\partial \eta^2} \tag{2.1}$$
$$+d(\xi,\eta)\frac{\partial u}{\partial \xi} + e(\xi,\eta)\frac{\partial u}{\partial \eta} + h(\xi,\eta)u = F,$$

where $(\xi,\eta) \in \Omega$. In any region of Ω where $b^2 - 4ac < 0$, the PDE problem is classified as elliptic, and in any region where $b^2 - 4ac = 0$ the PDE problem is said to be parabolic. Also, when $b^2 - 4ac > 0$ the problem is said to be hyperbolic [74]. Elliptic type BVPs, treated in this thesis, are written in the compact form,

$$\mathcal{D}_n\left[u(\mathbf{x})\right] = F_n(\mathbf{x}), \quad \mathbf{x} \in \mathcal{I} \tag{2.2}$$

where $\mathbf{x} \in \mathcal{I} \subset \mathbb{R}^r$, $\mathcal{D}_n$ is the differential operator, and $F_n : \mathbb{R}^r \to \mathbb{R}$ is a forcing function or source/sink term. The above PDE is subject to the BC,

$$\mathcal{B}[u(\mathbf{x})] = f(\mathbf{x}), \quad \mathbf{x} \in \partial\mathcal{I} \tag{2.3}$$

where $\mathbf{B}(\cdot)$ is a linear differential operator of order less than $\mathcal{D}(\cdot)$, and $f : \mathbb{R}^r \to \mathbb{R}$.

Parabolic and hyperbolic type IBVPs are represented in compact form by,

$$\frac{\partial^k u}{\partial t^k}(\mathbf{x},t) = \mathcal{D}_n[u(\mathbf{x},t)], \quad \mathbf{x} \in \mathcal{I} \tag{2.4}$$

subject to linear BC,

$$\mathcal{B}[u(\mathbf{x},t)] = f(\mathbf{x},t), \quad \mathbf{x} \in \partial\mathcal{I} \tag{2.5}$$

and initial (or terminal) condition(s),

$$\frac{\partial^\ell u}{\partial t^\ell}(\mathbf{x},t_0) = h_\ell(\mathbf{x}), \quad \ell = 0,...,k-1, \quad \mathbf{x} \in \mathcal{I}. \tag{2.6}$$

where $h_\ell : \mathbb{R}^r \to \mathbb{R}$, and $t \in [t_0, t_f)$.

The CPROP approach offers a natural paradigm for solving ODEs and PDEs via ANNs, because the ANN can be adapted to satisfy the differential operator, while preserving the I/BCs. Furthermore, it allows for the equality constraints to be satisfied during repeated incremental sessions by introducing a sequence of objective functions e_n, $n = 1, 2, \ldots$. Thus, it can be easily extended to the adaptive solution of PDEs in non-stationary environments.

In the presence of a non-stationary environment, and over long time scales, the underlying dynamic process may be subject to change and, as a result, the differential operator and/or forcing function in (2.2) and (2.4) may also change. In this case, a solution of (2.2) or (2.4) may be required for a sequence of PDEs, all in the same form, represented by a sequence of functions $\{(\mathcal{D}_n, F_n) : \ n = 1, 2, \ldots\}$, where each pair of functions $(\mathcal{D}_n, F_n)$ defines one elliptic or parabolic PDE. Thus, in this thesis, each PDE problem is labeled by n, and solved incrementally by adapting the same ANN solution. It is assumed that the n^{th} PDE problem holds for a period of time ΔT that is much greater than the time required to obtain the ANN solution. Therefore, the next PDE problem, labeled by $(n+1)$, can be approached after the ANN has converged to an acceptable solution of the n^{th} PDE problem.

2.2 Background on Constrained Backpropagation (CPROP)

Classical backpropagation solves an unconstrained optimization problem involving the minimization of a scalar objective function $e : \mathbb{R}^M \to \mathbb{R}$, with respect to the

ANN weights $\mathbf{w} \in \mathbb{R}^M$. In supervised learning, e is formulated in terms of a training set composed of known input and output information, $\mathcal{T} = \{\mathbf{x}_k, \mathbf{y}_k\}_{k=1,2,\ldots}$, and e is a measure of the error between the ANN output and the corresponding output $\mathbf{y}_k$. In reinforcement learning, the training set is given by input values $\mathcal{T} = \{\mathbf{x}_k\}_{k=1,2,\ldots}$, and e is a performance measure.

As in classical backpropagation problems, CPROP seeks to minimize a scalar objective function $e : \mathbb{R}^M \to \mathbb{R}$ formulated from training set $\mathcal{T}_S$. However, CPROP performs the training subject to a set of equality constraints. These constraints preserve the ANN's ability to satisfy data used in prior training, $\mathcal{T}_L$ or long term memory (LTM), at all times while training over new data, $\mathcal{T}_S$ or short term memory (STM). In the application of CPROP to PDEs the equality constraints arise from the boundary or initial conditions. These equality constraints are an example of a smooth function approximation that is solved using algebraic training [32] and are preserved via CPROP during subsequent incremental training sessions [31].

In previous work, CPROP has been used to eliminate interference and preserve prior knowledge in fully-connected sigmoidal neural networks, and to adapt an ANN-based nonlinear controller online, subject to changing and unmodeled aircraft dynamics [31]. In [26], CPROP was demonstrated on benchmark problems in function approximation, system identification, and the solution of ordinary differential equations.

After training an ANN over $\mathcal{T}_L$, algebraic training [32] is used to embed $\mathcal{T}_L$ into a functional relationship describing the network weights,

$$\mathbf{g}(\mathbf{w}_L, \mathbf{w}_S, \mathbf{x}_\ell) = \mathbf{0}, \quad \mathbf{x}_\ell \in \mathcal{T}_L \tag{2.7}$$

where the network weights have been partitioned into two vectors, $\mathbf{w}_L \in \mathbb{R}^{M_L}$ and $\mathbf{w}_S \in \mathbb{R}^{M_S}$. Then, a training method that preserves $\mathcal{T}_L$ while minimizing e can be

formulated as a constrained optimization problem:

$$\text{minimize} \quad e(\mathbf{w}_L, \mathbf{w}_S, \mathbf{x}_s), \quad \mathbf{x}_s \in \mathcal{T}_S \tag{2.8}$$

$$\text{subject to} \quad \mathbf{g}(\mathbf{w}_L, \mathbf{w}_S, \mathbf{x}_\ell) = \mathbf{0}, \quad \mathbf{x}_\ell \in \mathcal{T}_L. \tag{2.9}$$

Now, if (2.7) satisfies the implicit function theorem, then it uniquely implies the function,

$$\mathbf{w}_L = \mathcal{C}(\mathbf{w}_S) \tag{2.10}$$

and the method of direct elimination can be applied by re-writing the objective function as,

$$E(\mathbf{w}_S) = e(\mathcal{C}(\mathbf{w}_S), \mathbf{w}_S) \tag{2.11}$$

such that the value of $\mathbf{w}_S$ can be determined independently of $\mathbf{w}_L$. In this case, the solution of (2.8) is an extremum of (2.11) that obeys $\partial E/\partial \mathbf{w}_{S_{(j)}} = \mathbf{0}$ for $j = 1, ..., M_S$. Throughout this thesis the j^{th} element of a vector is denoted by a subscript (j). The j^{th} column of a matrix also is denoted by a subscript (j), and the element in the i^{th} row and j^{th} column of a matrix is denoted by a subscript (i, j).

Once the optimal value of $\mathbf{w}_S$ is determined, the optimal value of $\mathbf{w}_L$ can be obtained from $\mathbf{w}_S$ using (2.10). Furthermore, by use of the chain rule the *adjoined error gradient* is given by [113],

$$\frac{\partial E}{\partial \mathbf{w}_{S_{(i)}}} = \frac{\partial e}{\partial \mathbf{w}_{S_{(i)}}} + \frac{\partial e}{\partial \mathcal{C}} \frac{\partial \mathcal{C}}{\partial \mathbf{w}_{S_{(i)}}} \tag{2.12}$$

and the objective function can be written as

$$e(\mathbf{w}_L, \mathbf{w}_S) = \frac{1}{2} \boldsymbol{\epsilon}^T \boldsymbol{\epsilon} \tag{2.13}$$

where $\boldsymbol{\epsilon}_{(j)}$ is the error associated with the j^{th} point in the training set $\mathcal{T}_S$.

In this thesis, Levenberg-Marquardt (LM) is the training algorithm of choice because of its excellent convergence and stability properties [69, 62]. In the LM

algorithm, the update to the weights, $\Delta \mathbf{w}_S$, is found by solving a nonlinear system of equations,

$$\left(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I}\right)\Delta \mathbf{w}_S = -\mathbf{J}^T\boldsymbol{\epsilon} = -\nabla_{\mathbf{w}_S} E. \tag{2.14}$$

where, $\mathbf{I}$ is the identity matrix, μ is a positive constant, known as learning rate, and $\mathbf{J}$ is the Jacobian matrix. Then, a CPROP LM training algorithm can be obtained by deriving the adjoined Jacobian:

$$\begin{aligned} \mathbf{J}_{(m,n)}(\mathbf{w}_S) &= \frac{\partial \boldsymbol{\epsilon}_{(m)}[\mathcal{C}(\mathbf{w}_S), \mathbf{w}_S]}{\partial \mathbf{w}_{S_{(n)}}} = \frac{\partial \boldsymbol{\epsilon}_{(m)}(\mathbf{w}_L, \mathbf{w}_S)}{\partial \mathbf{w}_{S_{(n)}}} \\ &+ \frac{\partial \boldsymbol{\epsilon}_{(m)}[\mathcal{C}(\mathbf{w}_S), \mathbf{w}_S]}{\partial \mathcal{C}} \frac{\partial \mathcal{C}}{\partial \mathbf{w}_{S_{(n)}}} \end{aligned} \tag{2.15}$$

For the CPROP method, the ANN is chosen as a feedforward, one-hidden-layer, sigmoidal neural network, because of its universal function approximation ability [6, 63, 40]. The hidden layer can be represented by an operator with repeated sigmoidal functions, $\boldsymbol{\Phi}(\mathbf{n}) := [\sigma(n_1) \cdots \sigma(n_s)]^T$, where n_i denotes the i^{th} component of the input-to-node vector $\mathbf{n} \in \mathbb{R}^{s\times 1}$, and $\sigma(n_i) := (e^{n_i} - 1)/(e^{n_i} + 1)$. Then, the neural network input-output equation is,

$$\hat{y}(\mathbf{x}) = \boldsymbol{\Phi}(\mathbf{x}^T\mathbf{W}^T + \mathbf{b}^T)\mathbf{v}^T \tag{2.16}$$

where $\mathbf{b} \in \mathbb{R}^{s\times 1}$, $\mathbf{W} \in \mathbb{R}^{s\times r}$ and $\mathbf{v} \in \mathbb{R}^{1\times s}$, are the adjustable bias, input, and output weights, respectively.

From the CPROP equations (2.10)-(2.13), the method of direct elimination can be applied by partitioning the weights into an LTM set grouped in $\mathbf{w}_L$, and an STM set grouped in $\mathbf{w}_S$. The derivation of the adjoined derivatives can be simplified by partitioning the input weights, biases, and output weights into LTM and STM sets, denoted by subscripts L and S respectively, as shown in Fig. 2.1. Where, each set is identified by first partitioning hidden layer into 'S' nodes and 'L' nodes, and then

designating all weights connected to the 'S' nodes as STM weights, and all weights connected to the 'L' nodes as LTM weights. By this approach, the neural network input-output equation (2.16) can be re-written as,

$$\hat{y}(\mathbf{x}) = \mathbf{\Phi}\left(\mathbf{x}^T\mathbf{W}_L + \mathbf{b}_L\right)\mathbf{v}_L^T + \mathbf{\Phi}\left(\mathbf{x}^T\mathbf{W}_S + \mathbf{b}_S\right)\mathbf{v}_S^T \tag{2.17}$$

and used to derive the adjoined derivatives. In the following chapter, the derivation of the adjoined Jacobian (2.15), the objective function (2.13), and the explicit constraint equation (2.10) for the CPROP solution of elliptic and parabolic PDEs is given.

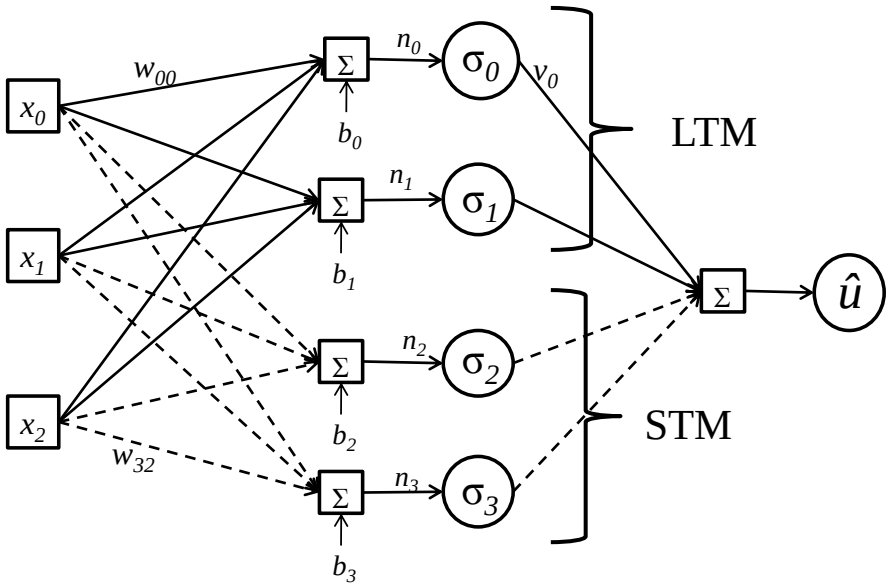


FIGURE 2.1: Partitioning of ANN nodes and weights.

2.3 Background on Galerkin Methods

The CINT method combines elements of CPROP (presented in the previous section) with Galerkin methods. Galerkin methods belong to the class of numerical methods for solving PDEs known as spectral methods, which approximate the solution to a PDE with a linear combination of basis functions. Spectral methods are widely implemented in various fields, including fluid dynamics, quantum mechanics, heat

conduction, and weather prediction [17, 36, 77, 46, 37]. This section presents a brief overview of spectral methods and their application to IBVPs. Spectral methods are not inherently adaptive, as the CPROP PDE solution method is, and so the subscript n on the differential operator in (2.4) will be dropped to indicate a stationary PDE.

In spectral methods, the PDE solution $u(\mathbf{x}, t)$ is approximated by a finite sum of linearly independent basis functions. Let $\{\phi_1(\mathbf{x}), ..., \phi_Q(\mathbf{x})\}$ denote a set of basis functions that belong to a Hilbert space with corresponding output weights $\mathbf{v}(t)$. Fourier and Chebyshev polynomials are commonly used bases due to the availability of the fast Fourier transformation (FFT) [53]. It is assumed that the approximate solution to (2.4) is given by

$$\hat{u}(\mathbf{x}, t) = \sum_{q}^{Q} \phi_q(\mathbf{x}) \mathbf{v}_{(q)}(t). \tag{2.18}$$

Now, let the inner product of two functions, $p(\mathbf{x})$, $q(\mathbf{x}) \in L^2$ be denoted by $\langle p(\mathbf{x}), q(\mathbf{x}) \rangle$ and defined as

$$\langle p(\mathbf{x}), q(\mathbf{x}) \rangle \triangleq \int_{\mathcal{I}} p(\mathbf{x}) \overline{q(\mathbf{x})} d\mathbf{x}. \tag{2.19}$$

In Galerkin methods, the approximate solution (2.18) is substituted into (2.4), and the inner-product (2.19) is used to arrive at the system of ODEs,

$$\mathbf{A} \frac{\partial^k \mathbf{v}}{\partial t^k}(t) = \mathbf{b}[\mathbf{v}(t)], \tag{2.20}$$

where the matrix $\mathbf{A} \in \mathbb{R}^{Q \times Q}$ and vector $\mathbf{b}[\mathbf{v}(t)] \in \mathbb{R}^Q$ are defined as

$$\mathbf{A}_{(i,j)} \triangleq \langle \phi_j(\mathbf{x}), \phi_i(\mathbf{x}) \rangle = \int_{\mathcal{I}} \phi_j(\mathbf{x}) \overline{\phi_i(\mathbf{x})} d\mathbf{x}, \tag{2.21}$$

$$\mathbf{b}_{(i)} \triangleq \langle \mathcal{D}[\hat{u}(\mathbf{x}, t)], \phi_i(\mathbf{x}) \rangle = \int_{\mathcal{I}} \mathcal{D}[\hat{u}(\mathbf{x}, t)] \overline{\phi_i(\mathbf{x})} d\mathbf{x}. \tag{2.22}$$

The initial condition(s) to the system of ODEs (2.20) is given by

$$\mathbf{A} \frac{\partial^\ell \mathbf{v}}{\partial t^\ell}(0) = \mathbf{q}, \tag{2.23}$$

where

$$\mathbf{q}_{(i)} \triangleq \langle g_\ell(\mathbf{x}), \phi_i(\mathbf{x}) \rangle = \int_{\mathcal{I}} g_\ell(\mathbf{x}) \overline{\phi_i(\mathbf{x})} d\mathbf{x}. \tag{2.24}$$

The boundary condition (5.8) is enforced by performing integration by parts on the right-hand side of (2.20) [18]. In practice, computing the right-hand side of (2.20) can be very expensive [41]. In particular, if $\mathcal{D}[\hat{u}(\mathbf{x}, t)]$ contains non-constant coefficients or non-linear terms, then a convolution of sums must be computed.

To simplify the enforcement of the boundary condition and to avoid computing convoluted sums, a pseudo-spectral method is often used. In pseudo-spectral methods, the solution is approximated at a set of discrete points, such that $\hat{u}_{i,j} \approx u(\mathbf{x}_i, t_j)$. Then, at each time step, t_j, the approximate solution $\hat{u}_{i,j}$ is transformed to the output weights, $\mathbf{v}(t_j)$, in the spectral domain. The spatial derivatives of $\hat{u}_{i,j}$ are then found by evaluating the partial derivatives of (2.18) at the collocation points $(\mathbf{x}_i, t_j)$, and the right-hand side of (2.4) is computed. This is done at each time step, and the values of the approximate solution, $\hat{u}_{i,j}$, located at collocation points along the boundary are adjusted in order to satisfy the boundary condition (2.5). This transformation can be performed in $O(N \log N)$ computations by means of the FFT, where N is the number of collocation points, $\mathbf{x}_j$, in $\mathcal{I}$. However, when N is large, the integration step size, Δt, is severely restricted [53].

Alternatively, a basis can be chosen or constructed that satisfies the boundary condition at each time step. For example, a Fourier series can be used for the special case that $\mathcal{I}$ is a rectangular domain and u is periodic on the boundary. The following Chapter demonstrates how the approximate solution to (2.4) is constrained in the CINT method to approximately satisfy the boundary condition (5.8) at each time step. The CINT method is then demonstrated on three IBVPs. Computational speed and accuracy of the CINT method are compared with Matlab's FE solver.

3

Methodology

The background given in the previous chapter is used in this chapter to develop the CPROP adaptive PDE solver and CINT methods. The first section gives the CPROP PDE solution method, which is applicable to linear and nonlinear elliptic type BVPs with linear BCs and parabolic type IBVPs with Dirichlet type BCs. Following the CPROP PDE methodology, the CINT method is given. The CINT method is applicable to parabolic and hyperbolic type IBVPs with linear BCs.

3.1 The Adaptive Constrained Backpropagation (CPROP) Method

As the nature of elliptic BVPs differ from parabolic IBVPs, the adaptive CPROP method is slightly different for each type of PDEs. Thus, this section is divided into subsections describing the methodology for elliptic type equations and for parabolic type equations.

3.1.1 Elliptic Boundary Value Problems

Consider the elliptic BVP (2.2). The LTM information that is to be preserved during training by being embedded in the equality constraint (2.7) is specified by the BC

(2.3), specifically, the training set

$$\mathcal{T}_L = \{\mathbf{x}_\ell, h(\mathbf{x}_\ell)\}_{\ell=1,\ldots,N_L}. \tag{3.1}$$

From (2.17), we seek an approximate ANN solution to the elliptic BVP problem (2.2),(2.3) in the form,

$$\hat{u}(\mathbf{x}) = \mathbf{\Phi}\left(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T\right)\mathbf{v}_L^T + \mathbf{\Phi}\left(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T\right)\mathbf{v}_S^T. \tag{3.2}$$

where, $\mathbf{W}_S$, $\mathbf{W}_L$, $\mathbf{b}_S$, $\mathbf{b}_L$, $\mathbf{v}_S$ and $\mathbf{v}_L$ are the adjustable ANN parameters. The number of 'L' nodes is determined from $\mathcal{T}_L$ using algebraic training [31, 32],[30]. The number of 'S' nodes is determined heuristically, based on the size and complexity of $\mathcal{T}_S$.

The objective function (2.13) for elliptic type BVPs is obtained by applying the differential operator to the approximate ANN solution (3.2), and evaluating the resulting function at points in the training set $\mathcal{T}_S$, giving

$$\boldsymbol{\epsilon}_{(j)} = \{\mathcal{D}_n[\hat{u}(\mathbf{x})] - F_n(\mathbf{x})\}_{\mathbf{x}=\mathbf{x}_j\in\mathcal{T}_S}. \tag{3.3}$$

To evaluate the the above equation, the partial derivatives of $\hat{u}(\mathbf{x})$ with respect to the input variables, $\mathbf{x}$, are required. Consider the partial derivative,

$$\chi(\mathbf{x}) = \frac{\partial^\gamma u(\mathbf{x})}{\partial\mathbf{x}_{(1)}^{m_1}\ldots\partial\mathbf{x}_{(r)}^{m_r}}, \tag{3.4}$$

where $\gamma = m_1 + \ldots + m_r$. Let $\boldsymbol{\omega}_{S_j}$ represent a diagonal matrix of the j^{th} column of $\mathbf{W}_S$, and let

$$\mathbf{\Lambda}_S = \prod_{j=1}^{r}\boldsymbol{\omega}_{S_j}^{m_j}. \tag{3.5}$$

Similarly, $\mathbf{\Lambda}_L$ is a product of diagonal matrices taken from columns of $\mathbf{W}_S$. Then,

differentiating (3.2) with respect to the elements of $\mathbf{x}$,

$$\begin{aligned}\frac{\partial^{\gamma}\hat{u}(\mathbf{x})}{\partial\mathbf{x}_{(1)}^{m_1}...\partial\mathbf{x}_{(r)}^{m_r}} &\equiv \hat{\chi}(\mathbf{x}) = \mathbf{\Phi}^{\gamma}\left(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T\right)\mathbf{\Lambda}_L\mathbf{v}_L^T \\ &+ \mathbf{\Phi}^{\gamma}\left(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T\right)\mathbf{\Lambda}_S\mathbf{v}_S^T \end{aligned} \tag{3.6}$$

where $\mathbf{\Phi}^{\gamma}(\cdot)$ denotes the γ^{th} derivative of the sigmoidal operator. Thus, all partial derivatives in the PDE differential operator $\mathcal{D}_n[\hat{u}(\mathbf{x})]$ can be derived in closed form from (3.2)-(3.5), and substituted in (3.3) to complete the expression for the ANN objective function (2.13).

The derives the ANN equality constraints and adjoined Jacobian that circumvent the need for substituting the equality constraints in the objective function directly are derived. Although the Jacobian depends on the form of the differential operator $\mathcal{D}_n(\cdot)$, its derivation can be illustrated through the partial derivative $\hat{\chi}$, defined in (3.4). Since the CPROP equations express equality constraints in terms of the PDE parameters and external forcing, the ANN solution can be adapted incrementally over time, to continue to satisfy parameters and/or external forcing that change as a result of non-stationary environments.

Because the forcing function F_n in the elliptic PDE (2.2) is independent of $\mathbf{w}_S$, it follows that $\partial\boldsymbol{\epsilon}_{(k)}/\partial\mathbf{w}_S = \partial\mathcal{D}_n[\hat{u}(\cdot)]/\partial\mathbf{w}_S|_{\mathbf{x}_k}$ for any k. Let M denote the number of partial derivatives of $\hat{u}$ in $\mathcal{D}_n[\hat{u}(\cdot)]$. Then, the equality constraint $\mathcal{D}_n[\hat{u}(\cdot)] = G(\hat{\chi}_1, ..., \hat{\chi}_M)$, the adjoined error gradient (2.12), and the Jacobian (2.15) can be obtained from the gradient

$$\frac{\partial\mathcal{D}_n[\hat{u}(\mathbf{x})]}{\partial\mathbf{w}_{S_{(k)}}} = \sum_i \frac{\partial G[\cdot]}{\partial\hat{\chi}_i}\frac{\partial\hat{\chi}_i}{\partial\mathbf{w}_{S_{(k)}}} \tag{3.7}$$

For every ith derivative, let

$$\frac{\partial G[\cdot]}{\partial\hat{\chi}_i}\frac{\partial\hat{\chi}_i}{\partial\mathbf{w}_{S_{(k)}}} = \frac{\partial G[\cdot]}{\partial\hat{\chi}_i}\left[\xi_1(\mathbf{x}) + \xi_2(\mathbf{x})\right] \tag{3.8}$$

where,

$$\begin{aligned}\xi_1(\mathbf{x}) &= \frac{\partial}{\partial \mathbf{w}_{S_{(k)}}}\left[\mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T)\mathbf{\Lambda}_L\mathbf{v}_L^T\right] \\ \xi_2(\mathbf{x}) &= \frac{\partial}{\partial \mathbf{w}_{S_{(k)}}}\left[\mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\mathbf{\Lambda}_S\mathbf{v}_S^T\right] \end{aligned} \tag{3.9}$$

and $\partial[\cdot]/\partial\mathbf{w}_{S_{(k)}}$ denotes the k^{th} element of the gradient vector $\partial[\cdot]/\partial\mathbf{w}_S$. Since $\mathbf{w}_S$ is obtained by re-grouping the elements of $\mathbf{W}_S$, $\mathbf{b}_S$, and $\mathbf{v}_S$, the partial derivatives with respect to these weights are derived separately as follows.

As a first step, consider the input weights, where $\mathbf{w}_{S_{(k)}}$ corresponds to the input weight $\mathbf{W}_{S_{(i,j)}}$, and let

$$\alpha_{ij} \equiv m_j \left(\mathbf{W}_{S_{(i,j)}}\right)^{m_j-1} \prod_{k\neq j}^{r} \left(\mathbf{W}_{S_{(i,k)}}\right)^{m_k}. \tag{3.10}$$

Then, for any input weight $\mathbf{W}_{S_{(i,j)}}$, the term $\xi_2(\mathbf{x})$ in (3.8) can be written as,

$$\begin{aligned}\xi_2(\mathbf{x}) &= \frac{\partial}{\partial \mathbf{W}_{S_{(i,j)}}}\left[\mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\mathbf{\Lambda}_S\mathbf{v}_S^T\right] \\ &= \Big[\alpha_{ij}\mathbf{\Phi}_{(i)}^{\gamma}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T) + \\ &\quad \mathbf{\Lambda}_{S_{(i,i)}}\mathbf{\Phi}_{(i)}^{\gamma+1}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\mathbf{x}_{(j)}\Big]\mathbf{v}_{S_{(i)}}\end{aligned} \tag{3.11}$$

and for any input bias $\mathbf{b}_{S_{(i)}}$, or output weight $\mathbf{v}_{S_{(i)}}$, $\xi_2(\mathbf{x})$ can be written as,

$$\begin{aligned}\xi_2(\mathbf{x}) &= \frac{\partial}{\partial \mathbf{b}_{S_{(i)}}}\left[\mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\mathbf{\Lambda}_S\mathbf{v}_S^T\right] \\ &= \mathbf{\Lambda}_{S_{(i,i)}}\mathbf{\Phi}_{(i)}^{\gamma+1}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\mathbf{v}_{S_{(i)}}\end{aligned} \tag{3.12}$$

or,

$$\begin{aligned}\xi_2(\mathbf{x}) &= \frac{\partial}{\partial \mathbf{v}_{S_{(i)}}}\left[\mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\mathbf{\Lambda}_S\mathbf{v}_S^T\right] \\ &= \mathbf{\Lambda}_{S_{(i,i)}}\mathbf{\Phi}_{(i)}^{\gamma}(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T)\end{aligned} \tag{3.13}$$

respectively. These equations provide the first term of the adjoined error gradient and Jacobian (2.12) and (2.15), and corresponds to the same partial derivatives used in classical backpropagation. The second term in the adjoined Jacobian (2.15) is given by the unused term in (3.8), $\partial F/\partial \hat{\chi}_i \times \xi_1(\mathbf{x})$. This term is subject to the constraint (2.10), and is derived as follows.

For an elliptic differential operator, the equality constraint is specified by a training set $\mathcal{T}_L$ defined from the BCs. When the BCs in (2.3) are imposed on the ANN approximate solution (3.2), they can be written as,

$$\begin{aligned} \mathcal{B}[\hat{u}(\mathbf{x})] &= \mathcal{B}\left[\mathbf{\Phi}\left(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T\right)\right]\mathbf{v}_L^T \\ &+ \mathcal{B}\left[\mathbf{\Phi}\left(\mathbf{x}^T\mathbf{W}_S^T + \mathbf{b}_S^T\right)\right]\mathbf{v}_S^T \end{aligned} \tag{3.14}$$

using the linearity of the operator $\mathcal{B}$. According to the algebraic training approach in [32], (3.14) is evaluated at the collocation points in the training set $\mathcal{T}_L$ and arranged into a linear system of equations. It follows that an ANN that satisfies $\mathcal{T}_L$ at all times can be obtained provided training satisfies the following equality constraint,

$$\mathbf{f} = \mathbf{\Psi}\mathbf{v}_L^T + \mathbf{\Omega}\mathbf{v}_S^T \tag{3.15}$$

where,

$$\mathbf{f}_{(j)} \equiv f(\mathbf{x}_j) \tag{3.16}$$

$$\mathbf{\Psi}_{(j,k)} \equiv \mathcal{B}\left[\mathbf{\Phi}_{(k)}\left(\mathbf{x}_j^T\mathbf{W}_L^T + \mathbf{b}_L^T\right)\right] \tag{3.17}$$

$$\mathbf{\Omega}_{(j,k)} \equiv \mathcal{B}\left[\mathbf{\Phi}_{(k)}\left(\mathbf{x}_j^T\mathbf{W}_S^T + \mathbf{b}_S^T\right)\right] \tag{3.18}$$

for all $\mathbf{x}_j \in \mathcal{T}_L$. Then, an explicit equality constraint in the form (2.10) can be obtained from (3.15) as follows,

$$\mathbf{v}_L^T = \mathbf{\Psi}^{-1}[\mathbf{f} - \mathbf{\Omega}\mathbf{v}_S^T], \tag{3.19}$$

where $\mathbf{\Psi}$ is assumed to be an invertible matrix that can be constructed using the method in [32].

According to the CPROP training approach reviewed in Section 2.2, the objective function (2.13) is minimized with respect to $\mathbf{w}_S$, while $\mathbf{W}_L$ is held constant. Thus, the matrix $\mathbf{\Psi}$ remains known and constant at all times. With the constraint now defined, the function $\xi_1(\mathbf{x})$ in (3.8) is

$$\xi_1(\mathbf{x}) = \mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T)\mathbf{\Lambda}_L\frac{\partial \mathbf{v}_L^T}{\partial \mathbf{w}_{S_{(k)}}}. \tag{3.20}$$

Then, when $\mathbf{w}_{S_{(k)}}$ corresponds to the input weight $\mathbf{W}_{S_{(\iota,m)}}$, the derivative of the constraint is given by

$$\frac{\partial \mathbf{v}_L^T}{\partial \mathbf{W}_{S_{(\iota,m)}}} = -\mathbf{\Psi}^{-1}\mathbf{y}\mathbf{v}_{S_{(\iota)}}. \tag{3.21}$$

For the points $\mathbf{x}_j \in \partial\mathcal{I}$ at which $\mathcal{B}$ defines Dirichlet conditions, the vector $\mathbf{y}$ in (3.21) is given by,

$$\mathbf{y}_{(j)} = \mathbf{x}_{j_{(m)}}\mathbf{\Phi}'_{(\iota)}(\mathbf{x}_j^T\mathbf{W}_S^T + \mathbf{b}_S^T). \tag{3.22}$$

whereas for points at which $\mathcal{B}$ defines BCs on the derivatives, this vector is given by,

$$\begin{aligned} \mathbf{y}_{(j)} &= \alpha_{\iota m}\mathbf{\Phi}^{\gamma}_{(\iota)}(\mathbf{x}_j^T\mathbf{W}_S^T + \mathbf{b}_S^T) \\ &+ \mathbf{\Lambda}_{S_{(\iota,\iota)}}\mathbf{x}_{j_{(m)}}\mathbf{\Phi}^{\gamma+1}_{(\iota)}(\mathbf{x}_j^T\mathbf{W}_S^T + \mathbf{b}_S^T). \end{aligned} \tag{3.23}$$

Similarly, for the input bias, the derivative of the constraint is given by

$$\frac{\partial \mathbf{v}_L^T}{\partial \mathbf{b}_{S_{(\iota)}}} = -\mathbf{\Psi}^{-1}\mathbf{y}\mathbf{v}_{S_{(\iota)}} \tag{3.24}$$

Where, for points at which $\mathcal{B}$ defines BCs Dirichlet boundary conditions, the vector $\mathbf{y}$ in (3.24) is,

$$\mathbf{y}_{(j)} = \mathbf{\Phi}'_{(\iota)}(\mathbf{x}_j^T\mathbf{W}_S^T + \mathbf{b}_S^T), \tag{3.25}$$

and for points at which $\mathcal{B}$ defines BCs on the derivatives

$$\mathbf{y}_{(j)} = \mathbf{\Lambda}_{S_{(\iota,\iota)}}\mathbf{\Phi}^{\gamma+1}_{(\iota)}(\mathbf{x}_j^T\mathbf{W}_S^T + \mathbf{b}_S^T). \tag{3.26}$$

Finally, the constrained derivatives for the output weights is,

$$\frac{\partial \mathbf{v}_L^T}{\partial \mathbf{v}_{S_{(\iota)}}} = -\mathbf{\Psi}^{-1}\mathbf{y}, \tag{3.27}$$

where,

$$\mathbf{y}_{(j)} = \mathbf{\Phi}_{(\iota)}(\mathbf{x}_j^T \mathbf{W}_S^T + \mathbf{b}_S^T), \tag{3.28}$$

for points with Dirichlet BCs, and,

$$\mathbf{y}_{(j)} = \mathbf{\Lambda}_{S_{(\iota,\iota)}} \mathbf{\Phi}_{(\iota)}^{\gamma}(\mathbf{x}_j^T \mathbf{W}_S^T + \mathbf{b}_S^T). \tag{3.29}$$

for points with BCs on the derivatives.

Equations (3.11)-(3.13) with (3.21)-(3.27) complete the derivation of the equality constraints and corresponding adjoined Jacobian for the elliptic BVP. The adjoined Jacobian, together with the unconstrained derivatives derived in this section, are then implemented by the CPROP algorithm to train the ANN [31]. The following subsection derives the constraints and adjoined Jacobian for parabolic IBVPs.

3.1.2 Parabolic Initial-Boundary Value Problems

Now, consider PDEs of the form given in (2.4). In order to simplify notation, the input variable t is assumed to be the r^{th} component of $\mathbf{x}$, such that $\mathbf{x} \in \mathcal{I}$, $\mathcal{I} = \mathcal{H} \times [t_0, t_f) \subset \mathbb{R}^r$, and $\mathcal{H}$ is a compact set. It is further assumed that the PDE (2.4) is of parabolic type, i.e. $k = 1$, and the differential operator, $\mathcal{B}(\cdot)$, in the BC (2.5) is the identity operator, giving a Dirichlet BC,

$$u(\mathbf{x}) = f(\mathbf{x}), \quad \forall \mathbf{x} \in \partial\mathcal{H} \times [t_0, t_f) \tag{3.30}$$

where $f : \mathbb{R}^r \to \mathbb{R}$. IBVPs differ from BVPs in that they also have an initial condition (2.6) associated with $\mathbf{x}_{(r)}$, where $t_0 \leqslant \mathbf{x}_{(r)} \leqslant t_f$. As it is assumed that the subscripts k and ℓ in (2.4) and (2.6) are fixed ($k = 1$, $\ell = 0$), these subscripts will be omitted in this section.

Based on the seminal work in [55, 70], the approximate ANN solution can be written as,

$$\begin{aligned} \hat{u}(\mathbf{x}) &= \tilde{f}(\mathbf{x}) + q(\mathbf{x}) \left[\mathbf{\Phi} \left(\mathbf{x}^T \mathbf{W}_L^T + \mathbf{b}_L^T \right) \mathbf{v}_L^T \right. \\ &+ \left. \mathbf{\Phi} \left(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{b}_S^T \right) \mathbf{v}_S^T \right] \end{aligned} \tag{3.31}$$

where $\tilde{f}(\mathbf{x})$ is differentiable for all $\mathbf{x}$ in $\mathcal{I}$, and satisfies the BC (3.30). The function $q : \mathbb{R}^r \rightarrow \mathbb{R}$ also is differentiable, and it is equal to zero along $\partial\mathcal{H}$, and nonzero in the interior of $\mathcal{H}$. When the BCs do not change over time, using (3.31) has the advantage that the BCs are automatically satisfied, leaving the initial condition as the only equality constraint.

The ANN objective function (2.13) to be minimized is obtained, similarly to the elliptic case, by applying the the parabolic differential operator in (2.4) to the approximate solution (3.31),

$$\boldsymbol{\epsilon}_{(j)} = \left\{ \frac{\partial \hat{u}(\mathbf{x})}{\partial \mathbf{x}_{(r)}} - \mathcal{D}_n[\hat{u}(\mathbf{x})] \right\}_{\mathbf{x}=\mathbf{x}_j \in \mathcal{T}_S}. \tag{3.32}$$

However, for the parabolic IBVP described in this subsection, the partial derivatives differ from (3.6) because of the form of the ANN approximate solution (3.31). The derivatives of (3.31) consist of products of $q(\mathbf{x})$ and its derivatives with derivatives of the ANN, $\hat{\chi}(\mathbf{x})$, as found in (3.6). First order derivatives are given by

$$\begin{aligned} \frac{\partial \hat{u}(\mathbf{x})}{\partial \mathbf{x}_{(j)}} &= \frac{\partial \tilde{h}(\mathbf{x})}{\partial \mathbf{x}_{(j)}} + \frac{\partial q(\mathbf{x})}{\partial \mathbf{x}_{(j)}} \left[\mathbf{\Phi} \left(\mathbf{x}^T \mathbf{W}_L^T + \mathbf{b}_L^T \right) \mathbf{v}_L^T \right. \\ &+ \left. \mathbf{\Phi} \left(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{b}_S^T \right) \mathbf{v}_S^T \right] + q(\mathbf{x}) \left[\mathbf{\Phi}^1 \left(\mathbf{x}^T \mathbf{W}_L^T \right. \right. \\ &+ \left. \left. \mathbf{b}_L^T \right) \boldsymbol{\omega}_{L_j} \mathbf{v}_L^T + \mathbf{\Phi}^1 \left(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{b}_S^T \right) \boldsymbol{\omega}_{S_j} \mathbf{v}_S^T \right] \end{aligned} \tag{3.33}$$

The second order derivatives of the approximate solution (3.31) is given by (A.1) in Appendix A.

With the above approximate solution structure (3.31), the ANN equality constraint (2.7) is obtained from the IC training set given by,

$$\begin{aligned}\mathcal{T}_L &= \{\mathbf{x}_\ell, \hat{h}(\mathbf{x}_\ell)\}_{\ell=1,\ldots,N_L} \\ &= \left\{ \left[\mathbf{x}_{(1)}, ..., \mathbf{x}_{(r-1)}, t_0\right]_\ell^T, \left.\frac{u(\mathbf{x}) - \tilde{f}(\mathbf{x})}{q(\mathbf{x})}\right|_{\mathbf{x}=\mathbf{x}_\ell} \right\}_{\ell=1,\ldots,N_L}\end{aligned}$$

for $\{\mathbf{x}_1, \ldots, \mathbf{x}_{r-1}\} \in \mathcal{H}$. The equality constraints are obtained by evaluating the ANN input-output equation in (3.31) at all points in the training set $\mathcal{T}_L$, defined in (3.34). The resulting set of algebraic equations are then organized into a linear system of equations that can be solved to obtain the explicit equality constraint,

$$\mathbf{v}_L^T = \mathbf{\Psi}^{-1}\left(\hat{\mathbf{h}} - \mathbf{\Omega}\mathbf{v}_S^T\right), \tag{3.34}$$

where $\mathbf{\Psi}$ and $\mathbf{\Omega}$ are defined as in the previous subsection, $\hat{\mathbf{h}}_{(j)} \equiv \hat{h}(\mathbf{x}_j)$, where $\hat{h}(\cdot)$ is defined in (3.34). It can be seen that if the problem is shifted so that $t_0 = 0$, the term $\mathbf{x}_k^T \mathbf{W}_L^T$ in (3.17) is independent of the weights in the r^{th} column of $\mathbf{W}_L$, and, thus, so is the equality constraint (3.34) representing the PDE ICs. This equality constraint is also independent of the r^{th} column of $\mathbf{W}_S$, and, thus, the corresponding derivatives needed to train these weights can be computed by means of classical backpropagation.

Since the equality constraint (3.34) is in the same form as the elliptic constraint (3.19), the derivatives in (3.21)-(3.27) are also used to compute the adjoined Jacobian for the parabolic IBVP. Then, given the explicit equality constraint equations, the adjoined Jacobian, and objective function, the CPROP algorithm can be used to determined the ANN weights incrementally, such that the chosen PDE is solved within a user-defined tolerance e_{tol}.

Table 3.1: Computational complexity of ANN PDE solution methods

	CPROP	FDM w/ ANN	Penalty Function
$\mathbf{J}$	$O(N_S N_L Q_S Q_L)/O(N_S N_L Q_S)$	$O(NQ)$	$O(NQ)$
$\mathbf{J}^T\mathbf{J}$	$O(N_S Q_S^2)$	$O(NQ^2)$	$O(NQ^2)$
LM update	$O(Q_S^3)$	$O(Q^3)$	$O(Q^3)$

3.1.3 Computational Complexity Analysis

A general concern is how fast the computational time grows with respect to the number of weights, which is proportional to the number of nodes in the hidden layer. Let Q_S represent the number of S nodes and Q_L the number of L nodes with $Q = Q_L + Q_S$ (Fig. 2.1). Also of interest is how the problem scales with the number of collocation points. Let N_S represent the cardinality of $\mathcal{T}_S$ and N_L the cardinality of $\mathcal{T}_L$, with $N = N_L + N_S$. The computational complexity of the adjoined Jacobian is derived here for the case of a linear elliptic/parabolic PDE. While it is difficult to derive the computational complexity of the adjoined Jacobian for nonlinear elliptic and parabolic PDEs, a useful comparison is made between the order of operations of the constrained and the unconstrained training algorithms. The results are summarized in Table 3.1.

Consider the equation for the derivatives of the differential operator in (3.8) needed to compute the adjoined gradient for the elliptic BVP using the CPROP algorithm. The term $\mathbf{\Phi}^\gamma(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T)$ is a row vector of length Q_L that is independent of $\mathbf{w}_S$. The derivative of $\mathbf{v}_L^T$ is given by (3.21) and (3.27), which multiply the matrix $\mathbf{\Psi}^{-1} \in \mathbb{R}^{Q_L \times N_L}$ by a column vector of length N_L. Then, the most computationally expensive operation is a matrix-vector multiplication, which is $O(Q_L N_L)$, and performing it for N_S collocation points and Q_S weights leads to a complexity of $O(Q_L N_L Q_S N_S)$ to compute the derivatives in (3.8) for the elliptic BVP. In the case of parabolic IBVP the complexity of (3.8) can be reduced compared to the elliptic

case, because $\mathbf{W}_L$ is held constant, and thus $\mathbf{\Phi}^{\gamma}(\mathbf{x}^T\mathbf{W}_L^T + \mathbf{b}_L^T)\mathbf{\Psi}^{-1}$ can be computed for all N_S points and stored prior to training. Then, the matrix-vector multiplication in (3.8) is reduced to a vector-vector multiplication with a total number of operations $O(N_L N_S Q_S)$.

The function $\xi_2(\mathbf{x})$ in (3.8) does not contain the constraint, and thus is of the same order as unconstrained LM, and the derivatives are given by equations (3.11)-(3.13). The most computationally expensive part of computing (3.11)-(3.13) is evaluating $\mathbf{\Phi}^{\gamma}$ at $N_S \times K_S$ points, which requires $O(N_S K_S)$ computations. When the differential operator is linear, $\partial \mathcal{D}_n[\hat{u}(\mathbf{x})]/\partial\hat{\chi}$ in in (3.8) is a constant, thus computing the Jacobian is $O(N_L N_S Q_S)$ for elliptic problems and $O(N_L N_S Q_L Q_S)$ for parabolic problems.

One approach that has been used extensively in the literature to solve PDEs via ANNs is to use a numerical solution method, such as FDM, to obtain a discrete solution in the form of a look-up table, and then to use this solution to train an ANN [87, 64]. Explicit FDM schemes are known to be $O(N)$, though in practice more points may be needed to obtain an accurate solution and avoid instabilities than required by the ANN CPROP solution. In this case, the most computationally expensive step in computing the (unconstrained) Jacobian is evaluating $\mathbf{\Phi}$ at NQ points, which is $O(NQ)$. It can be easily shown that this is also the computational cost of computing the Jacobian in penalty function methods, which essentially amount to including all collocation points in $\mathcal{T}_L$.

Assume the LM algorithm is used to train the ANN, either using classical unconstrained backpropagation or CPROP. LM requires computing $\mathbf{J}^T\mathbf{J}$, and solving the linear system of equations in (2.14) to update the weights (LM update). In the case of CPROP computing $\mathbf{J}^T\mathbf{J}$ is $O(N_S Q_S^2)$, while in the case of unconstrained ANN training (using the FDM solution or penalty function method), computing $\mathbf{J}^T\mathbf{J}$ is $O(NQ^2)$. The order of operations for solving the system of equations in (2.14) for

CPROP is $O(Q_S^3)$, while for the FDM solution and the penalty function method it is $O(Q^3)$.

As can be expected, the most expensive step in the CPROP method is computing the Jacobian, with complexity $O(N_S N_L Q_S)$ (for the elliptic case), while the most expensive step in the other methods is $O(NQ^2)$. Typically $N > Q$ to avoid over-fitting. Thus, it can be concluded that the computational complexity of CPROP is comparable both to the process of training an ANN using an FDM solution, and to the method of solving the PDE via ANNs using a penalty function to account for the BCs. Furthermore, CPROP eliminates the need for user intervention, as required by the FDM-based method, and requires less weights, less collocation points, and less training epochs the penalty function method, because it reduces the dimensionality of the optimization problem, as do all direct elimination methods. The numerical results presented in Chapter 4.1 demonstrate that the CPROP methodology can be used to solve elliptic BVPs and parabolic IBVPs adaptively with excellent accuracy.

The CPROP PDE method was expanded to the CINT method in order to decrease the computational time and improve accuracy in stationary IBVPs. The following section describes the CINT method, and compares it to Galerkin's method. The CINT and CPROP methods are demonstrated on several problems in Chapter 4

3.2 The Constrained Integration (CINT) Method

This section presents the CINT method. It is shown how the CPROP method is to preserve the BC, as well as how CINT compares to the classical Galerkin method. The primary differences between the two methods are in how they enforce the boundary condition in (5.8), and in how they approximate the integrals in (2.20)-(2.24) that are a result of the inner product. The linear combination of basis functions (2.18) is similar in structure to a feed-forward ANN with a single hidden layer. Using this paradigm, the boundary condition (5.8) is enforced in the CINT method using

a modification of the CPROP algorithm [31, 73].

Similarly to spectral methods, in the CINT method the solution to (2.4) is approximated by a feedforward ANN with a single hidden layer. As in [31, 73], the ANN is partitioned into two parts, one part is used to satisfy the BC (5.8) and the other to satisfy the PDE (2.4). The transfer functions used to preserve (5.8) are radial basis functions (RBFs) denoted by $\sigma_{L_1}(\mathbf{x}), ..., \sigma_{L_{Q_L}}(\mathbf{x})\}$. Gaussian RBFs are used for problems where the solution, $u(\mathbf{x}, t)$, is specified at the boundary (Dirichlet condition):

$$\sigma_{L_i}(\mathbf{x}) = \exp(-\gamma||\mathbf{x} - \mathbf{x}_i||^2). \tag{3.35}$$

In the above equation the shape parameter, γ, is a positive constant and $\mathbf{x}_i$ is a point along the boundary, around which the above RBF is centered. For PDE problems with a specified flux (Neumann condition), the transfer functions, $\sigma_{L_i}(\mathbf{x})$, are given by

$$\sigma_{L_i}(\mathbf{x}) = \exp(-\gamma||\mathbf{x} - \mathbf{x}_i||^2) \left(\frac{\exp[(\mathbf{x} - \mathbf{x}_i)^T \hat{\mathbf{n}}_i] - 1}{\exp[(\mathbf{x} - \mathbf{x}_i)^T \hat{\mathbf{n}}_i] + 1} \right), \tag{3.36}$$

where $\hat{\mathbf{n}}_i$ is the unit vector normal to the boundary at $\mathbf{x}_i$. The CINT transfer functions used to satisfy (2.4) are denoted by $\{\sigma_{S_1}(\mathbf{x}), ..., \sigma_{Q_L}(\mathbf{x})\}$. Polynomials and Fourier functions were found to work well in the IBVPs solved in Section 4.2. The ANN representation of the approximate solution of (2.4) is then given by

$$\begin{aligned} \hat{u}(\mathbf{x}, t) &= \sum_{i=1}^{Q_L} \sigma_{L_i}(\mathbf{x}) \mathbf{v}_{L_i}(t) + \sum_{j=1}^{Q_S} \sigma_{S_j}(\mathbf{x}) \mathbf{v}_{S_j}(t) \\ &= \boldsymbol{\sigma}_L^T(\mathbf{x}) \mathbf{v}_L(t) + \boldsymbol{\sigma}_S^T(\mathbf{x}) \mathbf{v}_S(t). \end{aligned} \tag{3.37}$$

To determine the above ANN solution, (3.37) is first substituted into (2.5), then the resulting equation is evaluated at a set of training or collocation points along the boundary, $\mathcal{T}_L = \{\mathbf{x}_k \mid \mathbf{x}_k \in \partial\mathcal{I}\}$. The boundary condition is then approximately

satisfied at these points provided

$$\mathbf{v}_L(t) = \mathbf{B}_L^+ \left[\mathbf{f}(t) - \mathbf{B}_S \mathbf{v}_S(t)\right], \tag{3.38}$$

where '+' denotes the pseudo-inverse, and

$$\mathbf{B}_{L_{(i,j)}} = \mathcal{B}[\sigma_{L_j}(\mathbf{x})]\big|_{\mathbf{x}=\mathbf{x}_i}, \tag{3.39}$$

$$\mathbf{B}_{S_{(i,j)}} = \mathcal{B}[\sigma_{S_j}(\mathbf{x})]\big|_{\mathbf{x}=\mathbf{x}_i}, \tag{3.40}$$

$$\mathbf{f}_{(i)}(t) = f(\mathbf{x}_i, t), \tag{3.41}$$

for $\mathbf{x}_i \in \mathcal{T}_L$. Substituting the right hand side of (3.38) for the output weights, $\mathbf{v}_L(t)$, in the ANN solution (3.37) yields an approximate solution that satisfies (5.8) to within some desired tolerance at each time step,

$$\begin{aligned}\hat{u}(\mathbf{x}, t) &= \left[\boldsymbol{\sigma}_S^T(\mathbf{x}) - \boldsymbol{\sigma}_L^T(\mathbf{x})\mathbf{B}_L^+\mathbf{B}_S\right]\mathbf{v}_S(t) + \boldsymbol{\sigma}_L^T(\mathbf{x})\mathbf{B}_L^+\mathbf{f}(t) \\ &= \sum_{i=1}^{Q_S} \phi_i(\mathbf{x})\mathbf{v}_{S_i}(t) + \zeta(\mathbf{x}, t).\end{aligned} \tag{3.42}$$

Rather than directly computing the inner product in (2.20), the constrained approximate solution (3.42) is substituted into the PDE in (2.4), and evaluated at a set of training or collocation points within the domain, $\mathcal{T}_S = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathcal{I}\}$, producing the following system of ODEs:

$$\mathbf{M}\frac{\partial^k \mathbf{v}_S}{\partial t^k}(t) = \boldsymbol{\xi}[\mathbf{v}_S(t)] \tag{3.43}$$

$$\mathbf{M}\frac{\partial^\ell \mathbf{v}_S}{\partial t^\ell}(t_0) = \mathbf{p}_\ell \tag{3.44}$$

where,

$$\mathbf{M}_{(i,j)} \triangleq \phi_j(\mathbf{x}_i), \tag{3.45}$$

$$\boldsymbol{\xi}_{(i)}[\mathbf{v}_S(t)] \triangleq \left\{\mathcal{D}[\hat{u}(\mathbf{x}, t)] - \frac{\partial^k \zeta}{\partial t^k}(\mathbf{x}, t)\right\}_{x=x_i}, \tag{3.46}$$

$$\mathbf{p}_{\ell_{(i)}} \triangleq g_\ell(\mathbf{x}_i) - \frac{\partial^\ell \zeta}{\partial t^\ell}(\mathbf{x}_i, t)\Big|_{t_0}. \tag{3.47}$$

We are now ready to state the main theoretical result for the CINT method.

Theorem 1. *If $\mathcal{D}[\hat{u}(\mathbf{x},t)]$ is Riemann integrable then the coefficients, $\mathbf{v}_S$, obtained by solving the systems that arise in Galerkin methods (2.20)-(2.24), are equal to those obtained by solving the linear least-squares problem described in (3.43) in the limit as the number of training points, N_S, goes to infinity.*

Proof. Let $\mathcal{I}$ be partitioned into N_S sub domains, $S_n \subset \mathcal{I}$, such that $m(S_n \cap S_k) = 0$, $\forall\, n \neq k$, and $m(S_n) = m(S_k) = \delta\mathbf{x}$, $\forall\, n, k \in \{1, ..., N_S\}$, where $m(S_n)$ is the measure of S_n. Let $\mathcal{T}_S$ be chosen such that $\forall \mathbf{x}_n \in \mathcal{T}$, $\mathbf{x}_n \in S_n$. Then the least squares solution to (3.43) is equal to the solution found by solving the equivalent system

$$\mathbf{M}^*\mathbf{M}\frac{\partial^k \mathbf{v}_S}{\partial t^k}\delta\mathbf{x} = \mathbf{M}^*\boldsymbol{\xi}\delta\mathbf{x} \tag{3.48}$$

where the superscript $*$ indicates the conjugate transpose. Now consider a single element of $\mathbf{M}^*\mathbf{M}$:

$$(\mathbf{M}^*\mathbf{M})_{(i,j)}\delta\mathbf{x} = \sum_{n=1}^{N_S} \phi_j(\mathbf{x}_n)\overline{\phi_i(\mathbf{x}_n)}\delta\mathbf{x} = \tag{3.49}$$

$$\sum_{n=1}^{N_S} \phi_j(\mathbf{x}_n)\overline{\phi_i(\mathbf{x}_n)}m(S_n).$$

The above equation is a Riemann sum, and as $N_S \to \infty$, (3.49) converges to the limit

$$\lim_{N_S\to\infty}(\mathbf{M}^*\mathbf{M})_{(i,j)}\delta\mathbf{x} = \tag{3.50}$$

$$\int_{\mathcal{I}} \phi_j(\mathbf{x})\overline{\phi_i(\mathbf{x})}d\mathbf{x} = \langle \phi_j(\mathbf{x}), \phi_i(\mathbf{x})\rangle = \mathbf{A}_{i,j}.$$

Similarly, consider a single element of the right-hand side of (3.48):

$$(\mathbf{M}^*\boldsymbol{\xi})_{(i)}\delta\mathbf{x} = \sum_{n=1}^{N_S}\left(\mathcal{D}[\hat{u}(\mathbf{x},t)] - \frac{\partial^k \zeta}{\partial t^k}(\mathbf{x},t)\right)\Bigg|_{\mathbf{x}_n} \overline{\phi_i(\mathbf{x}_n)}\delta\mathbf{x}$$

$$= \sum_{n=1}^{N_S}\left(\mathcal{D}[\hat{u}(\mathbf{x},t)] - \frac{\partial^k \zeta}{\partial t^k}(\mathbf{x},t)\right)\Bigg|_{\mathbf{x}_n} \overline{\phi_i(\mathbf{x}_n)}m(S_n). \tag{3.51}$$

The above equation is, again, a Riemann sum, and as $N_S \to \infty$, converges to the limit

$$\lim_{N_S \to \infty} (\mathbf{M}^* \boldsymbol{\xi})_{(i)} \delta \mathbf{x} = \tag{3.52}$$

$$\int_{\mathcal{I}} \left(\mathcal{D}[\hat{u}(\mathbf{x}, t)] - \frac{\partial^k \zeta}{\partial t^k}(\mathbf{x}, t) \right) \overline{\phi_i(\mathbf{x})} d\mathbf{x} =$$

$$\langle \mathcal{D}[\hat{u}(\mathbf{x}, t)] - \frac{\partial^k \zeta}{\partial t^k}(\mathbf{x}, t), \phi_i(\mathbf{x}) \rangle = \mathbf{b}_{(i)}.$$

□

As in pseudo-spectral schemes, in the CINT method the solution is transformed between real (u) and spectral ($\mathbf{v}_S$) spaces at each time step of the temporal integration. However, in pseudo-spectral methods the solution is transformed from real to spectral space where spatial derivatives are computed, then transformed back to real space. In the CINT method the coefficients, $\mathbf{v}_S$, are transformed to the approximate solution and its derivatives in real space via (2.18). The right-hand side of (2.4) is then evaluated and multiplied by $\mathbf{M}^+$ to perform an inverse transformation and obtain an approximation of the temporal derivatives of $\mathbf{v}_S(t)$.

In the following chapter the CINT method is demonstrated on three IBVPs. In the first two problems, the CINT method is applied to the wave equation in two spatial dimensions, which is a linear, hyperbolic PDE. In the final IBVP, the CINT method is used to solve the heat/diffusion equation in two spatial dimensions, which is a linear parabolic PDE. In the first two problems the CINT method outperforms the FE method both in terms of computational time and accuracy. For the parabolic heat/diffusion equation, the CINT and FE methods have similar performances.

4

Baseline Problems

In this chapter the adaptive CPROP and CINT PDE solution methods are demonstrated on several baseline PDE problems. The CPROP method is applied to a Poison problem with nonlinear forces, the heat/diffusion equation in two and three spatial dimensions, and the Boussinesq equation. The CINT method is demonstrated on the heat/diffusion equation and wave equation in two dimensions.

4.1 CPROP Numerical Simulations and Results

This section demonstrates the effectiveness of the CPROP methodology through several examples of elliptic and parabolic PDEs. Available methods of solution, such as the MATLAB® PDE Toolbox [2], are not applicable to all of the PDE problems considered in this section. Therefore, the CPROP solutions are compared to the best available numerical solution, on a case-by-case basis.

4.1.1 Adaptive CPROP Solution of Elliptic BVP

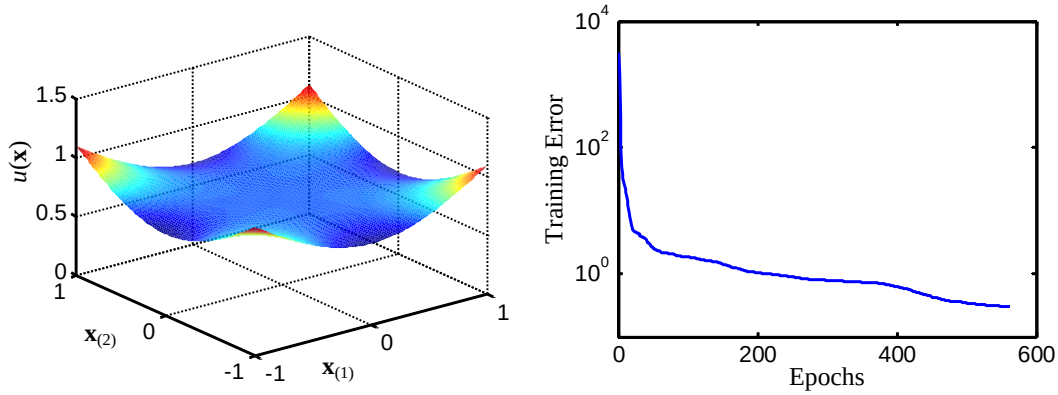
Consider the elliptic equation over the domain $\mathbf{x} \in \mathcal{I} = [-1, 1] \times [-1, 1]$,

$$\nabla^2 u(\mathbf{x}) + \alpha_n e^{u(\mathbf{x})} = \alpha_n \left[1 + \mathbf{x}_{(1)}^2 + \mathbf{x}_{(2)}^2 + \frac{4}{(1 + \mathbf{x}_{(1)}^2 + \mathbf{x}_{(2)}^2)^2} \right], \tag{4.1}$$

with the boundary condition

$$u(\mathbf{x}) = \log(\mathbf{x}_{(1)}^2 + \mathbf{x}_{(2)}^2 + 1), \quad \forall \mathbf{x} \in \partial\mathcal{I}. \tag{4.2}$$

The above PDE can be used to capture many dynamic processes in fluid mechanics, electrostatics, and thermodynamics, such as, steady incompressible irrotational fluid flow in two dimensions, and heat/diffusion processes in steady state. The effect of non-stationary environments is simulated by changing the parameter α_n, representing the relative importance of the nonlinear term versus the forcing function. A sequence of six PDEs problems in the form (4.1)-(4.2) is obtained by letting $n = 0, \ldots, 5$, and $\alpha_n = 0.2n$. For $\alpha_n = 0$, the PDE in (4.1) reduces to Laplace's equation.

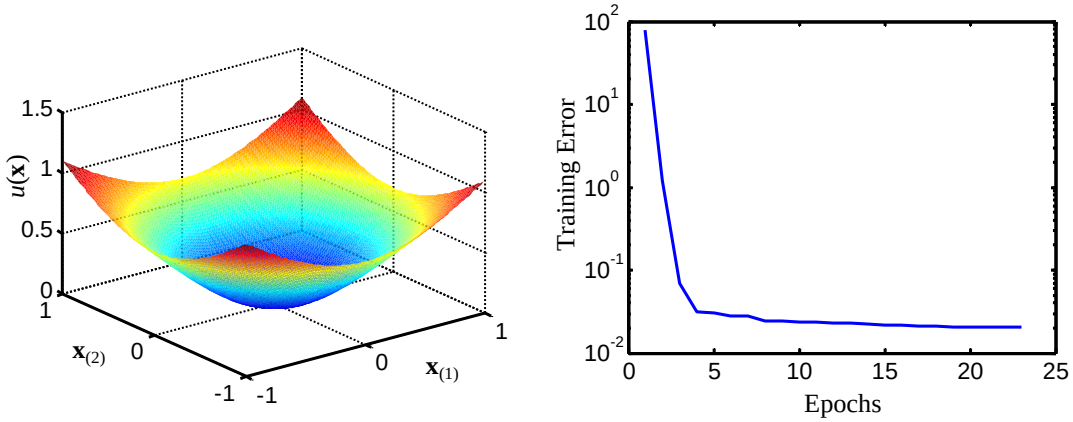


(a) (b)

FIGURE 4.1: CPROP solution to the elliptic PDE (4.1) when $n = 0$ (a), and corresponding training error (b).

Using the CPROP methodology presented in Section 3.1, the ANN in (3.2) is trained to solve these six PDEs adaptively. When the objective function decreases

below e_{tol}, the CPROP algorithm ceases training the ANN. Then, when the value of α_n is modified, the change is reflected in the training sets and, subsequently, in the objective function. As a result, the objective function exceeds e_{tol}, and the CPROP algorithm resumes training the ANN incrementally, starting with the weights obtained during the last training session. The input data in $\mathcal{T}_L$ consists of 180 equally spaced collocation points in $\partial\mathcal{I}$. The input data in $\mathcal{T}_S$ consists of a 35×35 grid of points in the interior of $\mathcal{I}$. The corresponding output data for the two training sets is computed as explained in Section 3.1.1. The ANN is partitioned into 40 'L' nodes and 20 'S' nodes (Fig. 2.1). The training set $\mathcal{T}_S$ is used to formulate the objective function (3.3) to be minimized in terms of the differential operator in (4.1).



(a) (b)

FIGURE 4.2: CPROP solution to the elliptic PDE (4.1) when $n = 5$ (a), and corresponding training error (b).

At $n = 0$, the weights are initialized randomly. No training of $\mathbf{W}_L$ and $\mathbf{b}_L$ is required and, instead, it is sufficient to initialize the input weights with uniformly distributed values in the interval $(-5, 5)$, similarly to [45, 65]. The CPROP adaptive solution is obtained for $n = 0, ..., 5$, and is shown in Figs. 4.1 and 4.2 for $n = 0$ and $n = 5$.

For $n = 0$, the CPROP solution is compared to the solution obtained using the

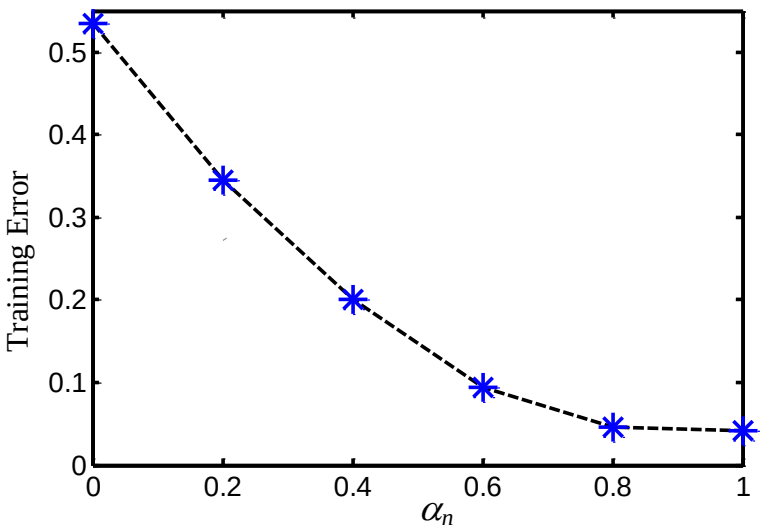


FIGURE 4.3: Final training error of elliptic PDEs CPROP solutions.

MATLAB® PDE Toolbox [2], and for $n = 5$ it is compared to the analytical solution $u(\mathbf{x}) = \log(\mathbf{x}_{(1)}^2 + \mathbf{x}_{(2)}^2 + 1)$ [70]. Plots of these solutions are not included for brevity, but the corresponding Relative Error Norm (REN),

$$E = \frac{\sum_i (u(\mathbf{x}_i) - \hat{u}(\mathbf{x}_i))^2}{\sum_i u^2(\mathbf{x}_i)} \tag{4.3}$$

computed using a validation set is plotted in Fig. 4.4 for the elliptic PDE (4.1) with $n = 5$, solved via CPROP 800 times. Each box plot shows the distribution of REN resulting from 100 simulations, using different numbers of nodes. Similar results were obtained for $n = 0$, but are omitted for brevity.

For $\alpha_n \neq 0$ the MATLAB® PDE Toolbox cannot be used to solve (4.1) due to the presence of the nonlinearity. Therefore, as a form of comparison the final training error associated with each PDE problem is plotted as a function of α_n in Fig. 4.3. It can be seen that the errors are similar for all value of α_n and, in fact, decreasing with each new adaptation of the solution. The history of the errors demonstrates that the CPROP methodology is benefiting from solving the PDE online, exploiting the previous PDE solution as an excellent initial estimate. Moreover, by this approach, a

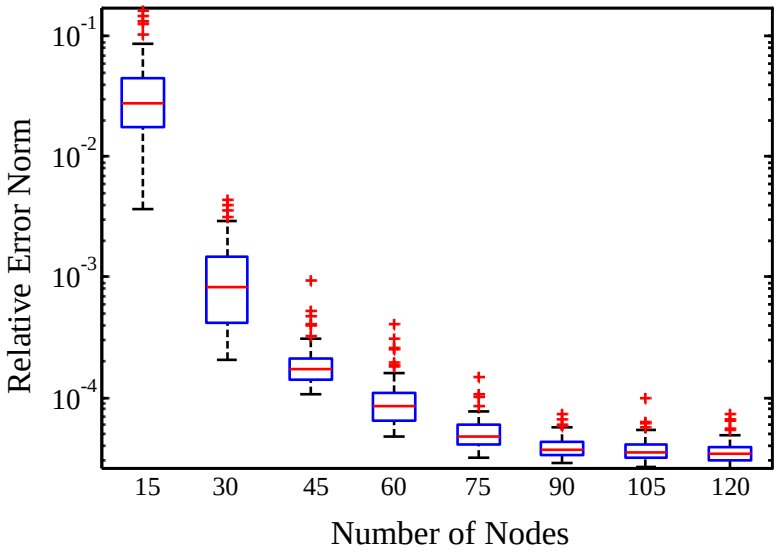


FIGURE 4.4: Box plots of REN vs. the number of nodes in the NN.

reasonable approximation to the PDE solution is available at all times in functional form. These plots also show that fewer iterations are required to converge to a satisfactory solution, compared to when the initial weights are initialized randomly. This result is verified in Fig. 4.5, where five box plots, each representing 100 solutions, show that the number of epochs required by the adaptive solution is far less than that required by the non-adaptive solution. As can be expected, the adaptive solution is most effective when the change between the $n-1$ and n equations are incremental, as if the difference is very large, then the computational savings are not significant.

4.1.2 Adaptive CPROP Solution of a two-dimensional linear, unsteady heat/diffusion IBVPs

This subsection presents the results obtained for a two-dimensional (2D) linear, unsteady, heat/diffusion equation without convection or source/sink terms, which is one of the most basic parabolic equations. The PDE problem is solved adaptively, subject to a changing coefficient that represents the diffusivity of the material.

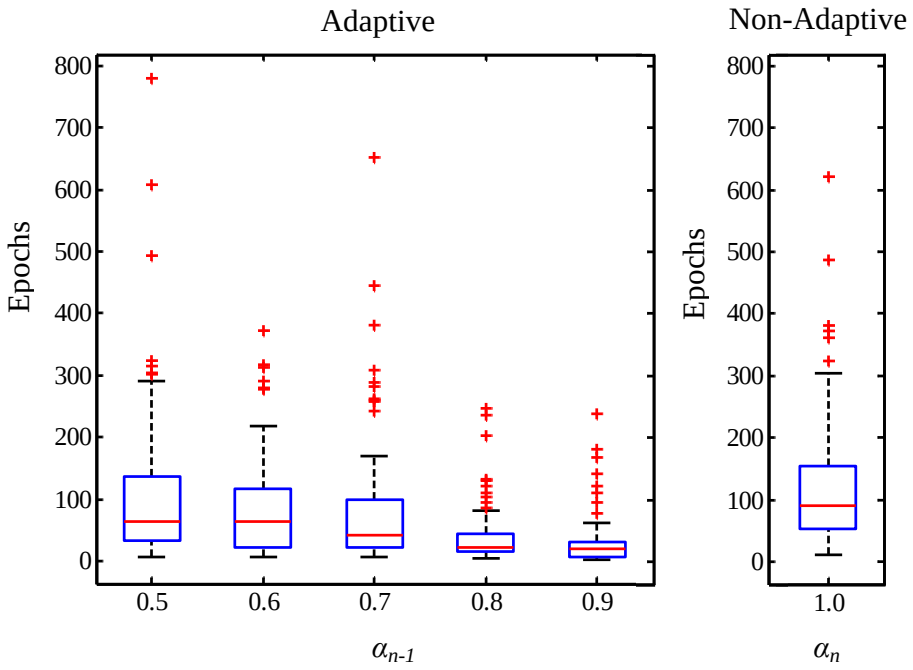


FIGURE 4.5: Box plots of CPROP training epochs needed to solve the elliptic PDE (4.1) adaptively from the α_{n-1} solution, and non-adaptively.

The unsteady, linear 2D heat/diffusion equation is,

$$\frac{\partial u(\mathbf{x})}{\partial \mathbf{x}_{(3)}} = k_n \left[\frac{\partial^2 u(\mathbf{x})}{\partial \mathbf{x}_{(1)}^2} + \frac{\partial^2 u(\mathbf{x})}{\partial \mathbf{x}_{(2)}^2} \right] \tag{4.4}$$

where $u(\mathbf{x})$ is represents the temperature in the heat equation, or density in the diffusion equation. The coefficient k_n, which is typically held constant, represents the diffusivity of the material, and determines the rate at which heat or mass is diffused through the system. The domain of the PDE is $(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \in \mathcal{H} = [-1, 1] \times [-1, 1]$, and $\mathbf{x}_{(3)} \geqslant 0$, where $\mathbf{x}_{(3)}$ represents time. The PDE in (4.4) is subject to Dirichlet boundary conditions,

$$u(\mathbf{x}) = 0, \quad \forall (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \in \partial\mathcal{H}, \tag{4.5}$$

and to the initial condition,

$$u(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, 0) = e^{-7(\mathbf{x}_{(1)}^2 + \mathbf{x}_{(2)}^2)} \sin(2\pi \mathbf{x}_{(1)}), \quad \forall \quad (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \in \mathcal{H}, \tag{4.6}$$

which specifies the solution everywhere in $\mathcal{H}$ at time zero.

The ANN solution takes the form (3.31), with a user-defined function $q(\mathbf{x}) \equiv (\mathbf{x}_{(1)}^2 - 1)(\mathbf{x}_{(2)}^2 - 1)$. The ANN in (3.31) is chosen to have 50 L nodes and 30 S nodes (Fig. 2.1). The input data in $\mathcal{T}_L$ consists of a 30×30 grid of equally spaced points in $\mathcal{H}$, which are used together with the ICs (4.6) to formulate the equality constraint 3.34. The input data in $\mathcal{T}_S$ consists of a $15 \times 15 \times 15$ lattice of points in $\mathcal{H} \times (0, 1]$.

To simulate the effects of non-stationary environments, two PDE problems in the form (4.4)-(4.6) were considered by letting $n = 0, 1$, with $k_0 = 0.01$, and $k_1 = 0.1$. The results in Fig. 4.6 show sample snapshots of the PDE solution obtained using MATLAB® and CPROP at sample moments in time. The CPROP training error (omitted for brevity) shows that, after an instantaneous increase due to the changing coefficient, the error decreases significantly until CPROP converges to optimal weight values. The adaptive solution is plotted in Fig. 4.8 for $n = 1$, and compared to the (non-adaptive) MATLAB®. It can be seen that the adaptive solution rapidly converges to the steady-state zero solution, despite the fact that this type of flat function is one of the hardest to approximate via ANNs. The REN of the parabolic PDE (4.4) obtained by solving the problem 160 times is plotted in Fig. 4.7. Each box plot represents the REN from 20 approximate solutions to the parabolic PDE for $n = 0$, showing that the accuracy can be improved by increasing the number of nodes. Similar results were obtained for $n = 1$ but are omitted for brevity.

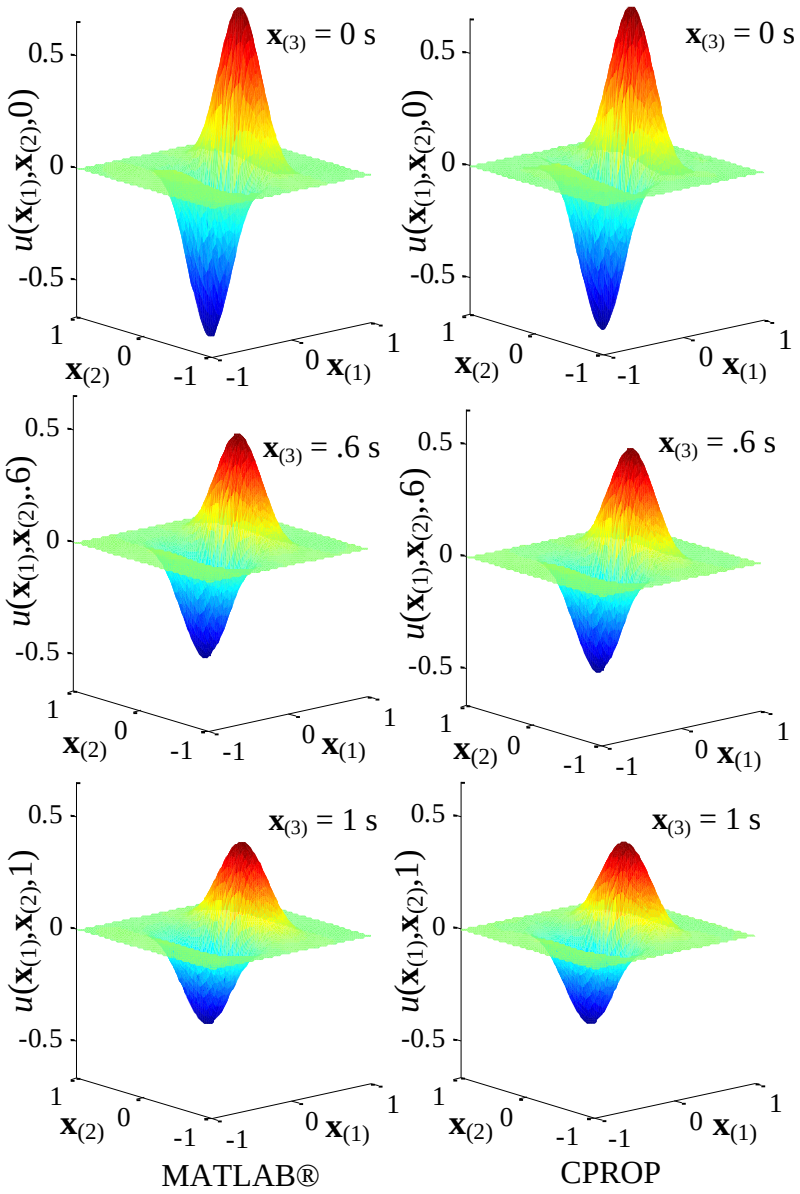


FIGURE 4.6: 2D-heat/diffusion equation solutions obtained using MATLAB® and CPROP when $n = 0$, $\mathbf{x}_{(3)} = 0$ s, $\mathbf{x}_{(3)} = 0.6$ s, and $\mathbf{x}_{(3)} = 1$ s.

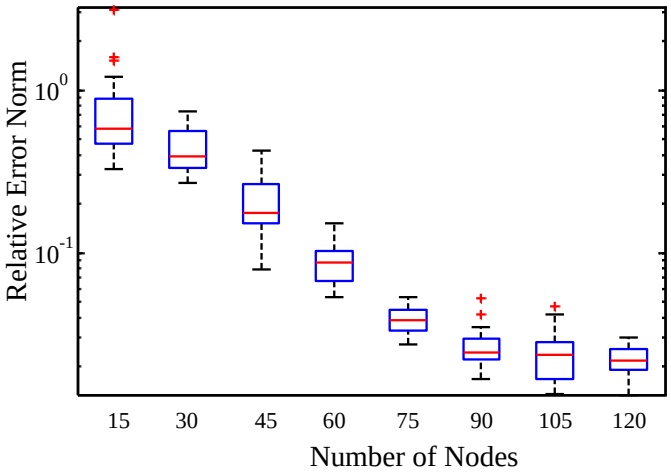


FIGURE 4.7: Box plots of relative error norm with respect to the number of nodes.

4.1.3 Adaptive CPROP Solution of a three-dimensional linear, unsteady heat/diffusion IBVPs

The unsteady, linear 3D heat/diffusion equation is,

$$\frac{\partial u(\mathbf{u})}{\partial \mathbf{x}_{(4)}} = k_n \left[\frac{\partial^2 u(\mathbf{x})}{\partial \mathbf{x}_{(1)}^2} + \frac{\partial^2 u(\mathbf{x})}{\partial \mathbf{x}_{(2)}^2} + \frac{\partial^2 u(\mathbf{x})}{\partial \mathbf{x}_{(3)}^2} \right] \tag{4.7}$$

where $(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \mathbf{x}_{(3)}) \in \mathcal{I} = [-1, 1] \times [-1, 1] \times [-1, 1]$, and $\mathbf{x}_{(4)} \geqslant 0$ represents time. The parabolic PDE in (4.7) is subject to the BCs,

$$u(\mathbf{x}) = 0, \quad \forall (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \mathbf{x}_{(3)}) \in \partial\mathcal{I}. \tag{4.8}$$

and to the ICs,

$$\begin{aligned} u(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \mathbf{x}_{(3)}, 0) &= 2\left(e^{-10||\mathbf{x}-\mathbf{x}_0||^2} - e^{-10||\mathbf{x}+\mathbf{x}_0||^2}\right) \\ \forall &\quad (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \mathbf{x}_{(3)}) \in \mathcal{I}, \end{aligned} \tag{4.9}$$

where $\mathbf{x}_0 = [0.5\ 0.5\ 0.5\ 0]^T$ is a known and constant vector.

The above PDE problem is chosen to demonstrate the CPROP method's ability to cope with several variables, and to adapt a 4D PDE solution to a non-stationary environment, by letting $n = 0, 1$, where $k_0 = 0.01$ and $k_1 = 0.1$. The ANN solution

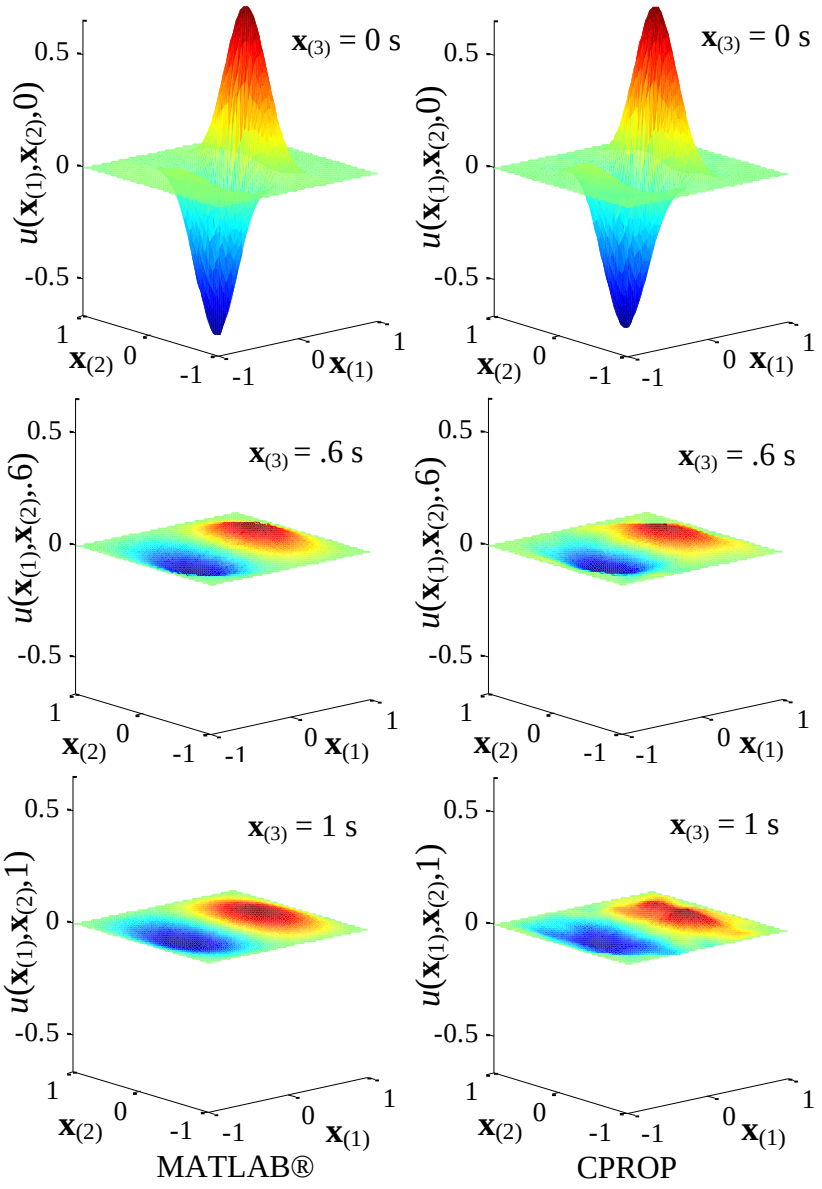


FIGURE 4.8: Adaptive CPROP solution for the 2D heat/diffusion (4.4) when $n = 1$ is compared to the (non-adaptive) solution obtained using MATLAB® at times $\mathbf{x}_{(3)} = 0$ s, $\mathbf{x}_{(3)} = 0.6$ s, and $\mathbf{x}_{(3)} = 1$ s.

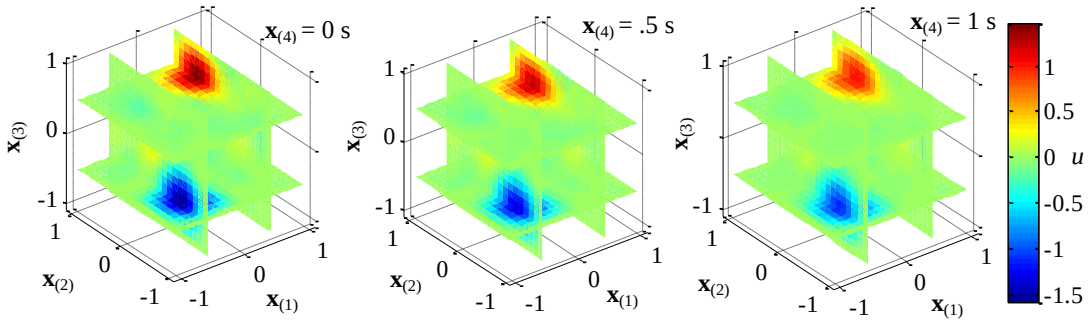


FIGURE 4.9: CPROP solution of the 3D heat/diffusion equation (4.7) when $n = 0$.

takes the form (3.31), with a user-defined function $q(\mathbf{x}) = (\mathbf{x}_{(1)}^2 - 1)(\mathbf{x}_{(2)}^2 - 1)(\mathbf{x}_{(3)}^2 - 1)$. The ANN architecture consists of 30 'L' nodes, and 60 'S' nodes. The input data in $\mathcal{T}_L$ consists of a $25 \times 25 \times 25$ lattice of spatial points in the interior of $\mathcal{I}$, while the input data in $\mathcal{T}_S$ consists of an $8 \times 8 \times 8 \times 8$ lattice of points in $\mathcal{H} \times (0, 1]$.

When $n = 0$, an accurate ANN PDE solution is obtained using CPROP, as shown in Fig. 4.9, using randomly initialized weights. For $n = 1$, the ANN PDE solution is adapted by the CPROP algorithm, as shown in Fig. 4.10. These solutions could not be compared to the MATLAB® solution, because the MATLAB® PDE Toolbox is not capable of solving 3D problems [2]. However, the CPROP solution is found to converge to the steady state solution, known to be $u = 0$, and the training errors (not shown for brevity) confirm convergence to optimal ANN weights. To illustrate the computational savings brought about by the adaptive CPROP solution, the 3D heat/diffusion equation (4.7) with $n = 1$ was solved 20 times using random initial weights (non-adaptively). The difference in number of epochs required by the adaptive and non-adaptive ANN solution is plotted in Fig. 4.11, where it can be seen that the former converges significantly faster than the latter.

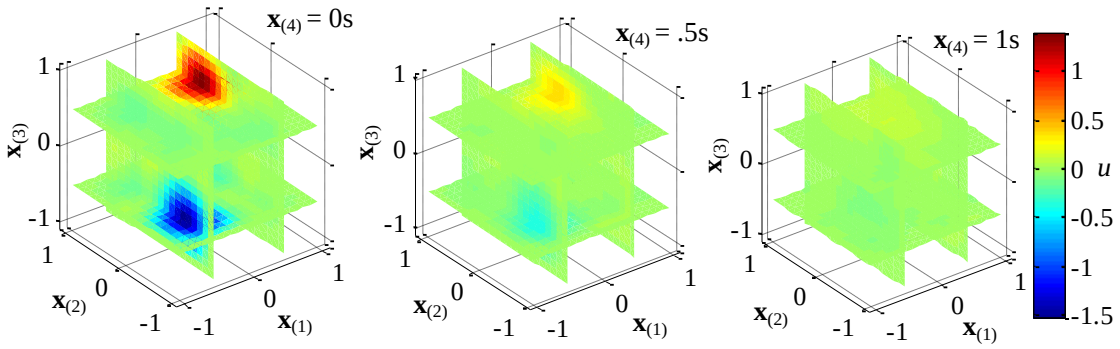


FIGURE 4.10: CPROP solution of the 3D heat/diffusion equation (4.7) when $n = 1$.

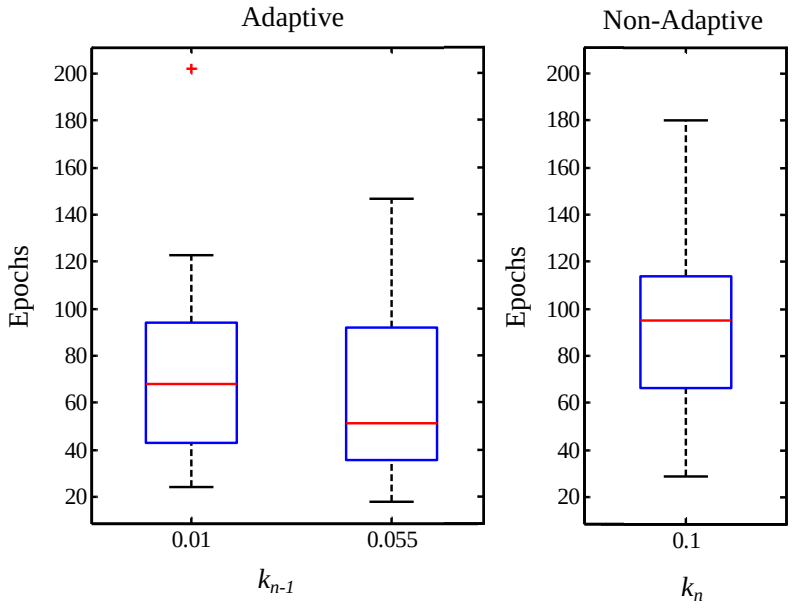


FIGURE 4.11: Box plots of number of CPROP training epochs needed to solve the 3D heat/diffusion equation (4.4) adaptively and non-adaptively.

4.1.4 Adaptive CPROP Solution of the Boussinesq Equation

The CPROP methodology is demonstrated on a nonlinear diffusion PDE problem, commonly known as the Boussinesq equation, which is chosen to show the applicability of the method to nonlinear, parabolic IBVPs. The Boussinesq equation is a model of heat/diffusion process with nonlinear diffusive properties that is used extensively

in numerical groundwater flow simulations [24], and can be written as,

$$S_n \frac{\partial u}{\partial \mathbf{x}_{(3)}} = \frac{\partial}{\partial \mathbf{x}_{(1)}} \left[K_n u \frac{\partial u}{\partial \mathbf{x}_{(1)}} \right] + \frac{\partial}{\partial \mathbf{x}_{(2)}} \left[K_n u \frac{\partial u}{\partial \mathbf{x}_{(2)}} \right] \tag{4.10}$$

where u is the elevation of the water table above a horizontal base, the spatial coordinates are $(\mathbf{x}_{(1)}, \mathbf{x}_{(\mathbf{2})}) \in \mathcal{H} = [-1, 1] \times [-1, 1]$, and $\mathbf{x}_{(3)} \geqslant 0$ is time. S_n is the specific yield, or the amount of water released per volume of porous medium when changing from a saturated state to an unsaturated state high above the water table, and K_n is the hydraulic conductivity.

In many applications K_n and S_n are assumed to be constant. To demonstrate the ability of CPROP to solve nonlinear parabolic IBVPs adaptively, in this paper the specific yield and hydraulic connectivity are modeled as,

$$S_n(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) = .2 \left(1 + \beta_n \frac{e^{10\mathbf{x}_{(1)} + 2\mathbf{x}_{(2)}} - 1}{e^{10\mathbf{x}_{(1)} + 2\mathbf{x}_{(2)}} + 1} \right) \tag{4.11}$$

$$K_n(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) = .0002 \left(1 - \beta_n \frac{e^{10\mathbf{x}_{(1)} + 2\mathbf{x}_{(2)}} - 1}{e^{10\mathbf{x}_{(1)} + 2\mathbf{x}_{(2)}} + 1} \right) \tag{4.12}$$

also rendering the PDE problem considerably more challenging. The Boussinesq PDE (4.10) is subject to the ICs,

$$\begin{aligned} u(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, 0) = 10 + 9 \sin(\pi(\mathbf{x}_{(1)}\mathbf{x}_{(2)} + \mathbf{x}_{(1)}^2)) \\ \times \cos(2\pi(\mathbf{x}_{(2)}^2 + .1))(\mathbf{x}_{(1)}^2 - 1)(\mathbf{x}_{(2)}^2 - 1) \\ \times \exp(-\sin^2(2\pi\mathbf{x}_{(2)})), \quad \forall (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \in \mathcal{H} \end{aligned} \tag{4.13}$$

and to the Dirichlet BCs

$$u(\mathbf{x}) = 10, \quad \forall (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \in \partial\mathcal{H} \tag{4.14}$$

Because MATLAB® PDE toolbox is only capable of solving linear parabolic PDEs, for comparison, the PDE problem in (4.10)-(4.14) was solved using FDM.

The finite difference scheme implemented in this paper discretizes the domain of the problem, letting,

$$u_{i,j}^{m} = u(-1 + i\Delta\mathbf{x}_{(1)}, -1 + j\Delta\mathbf{x}_{(2)}, m\Delta\mathbf{x}_{3}). \tag{4.15}$$

denote a point-wise solution, and using a forward stepping temporal difference to approximate $\partial u/\partial \mathbf{x}_{(3)}$ with central differencing for spatial derivatives. The stencil is shown in (B.1) in Appendix B. The domain $\mathcal{H}$ is discretized using $\Delta\mathbf{x}_{(3)} = 5e-4$, and $\Delta\mathbf{x}_{(1)} = \Delta\mathbf{x}_{(2)} = 0.02$. The Boussinesq equation being nonlinear makes it difficult to use implicit or semi-implicit schemes, such as Crank-Nicolson, that are known to exhibit higher stability than strictly explicit schemes, such as the one used in this paper. Thus $\Delta\mathbf{x}_{(3)}$ is chosen to be very small to avoid stability issues.

The CPROP solution to (4.10)-(4.14) is obtained using input data in $\mathcal{T}_L$ that consists of a 40×40 grid in $\mathcal{H}$, and input data in $\mathcal{T}_S$ that consists of a $20 \times 20 \times 20$ lattice in $\mathcal{H} \times (0, 1]$, with 110 'L' nodes and 30 'S' nodes. Compared to the previous examples, the IC in this example was more intricate. Thus a larger number of 'L' nodes as well as training points were required to obtain desired accuracy. In order to simulate the effect of non-stationary environments, the parameter β_n in the specific yield and hydraulic connectivity equations, (4.11) and (4.12), is varied from $\beta_0 = 0$ to $\beta_1 = 0.5$. As a result, two Boussinesq PDE problems are obtained for $n = 0$ and $n = 1$.

For $n = 0$, the ANN weights were initialized randomly, and the CPROP solution is plotted in Fig. 4.12 and compared to the FDM solution at different moments in time. When $n = 1$, the CPROP solution was adapted, and plotted in Fig. 4.13, along with the comparison with an FDM solution obtained by solving the same PDE problem non-adaptively. As in previous examples, the Boussinesq PDE problem with $\beta_n = 0.5$ was solved 20 times (non-adaptively) with FDM, and compared to the adaptive solutions. The resulting box plot (Fig. 4.14) illustrates that the adap-

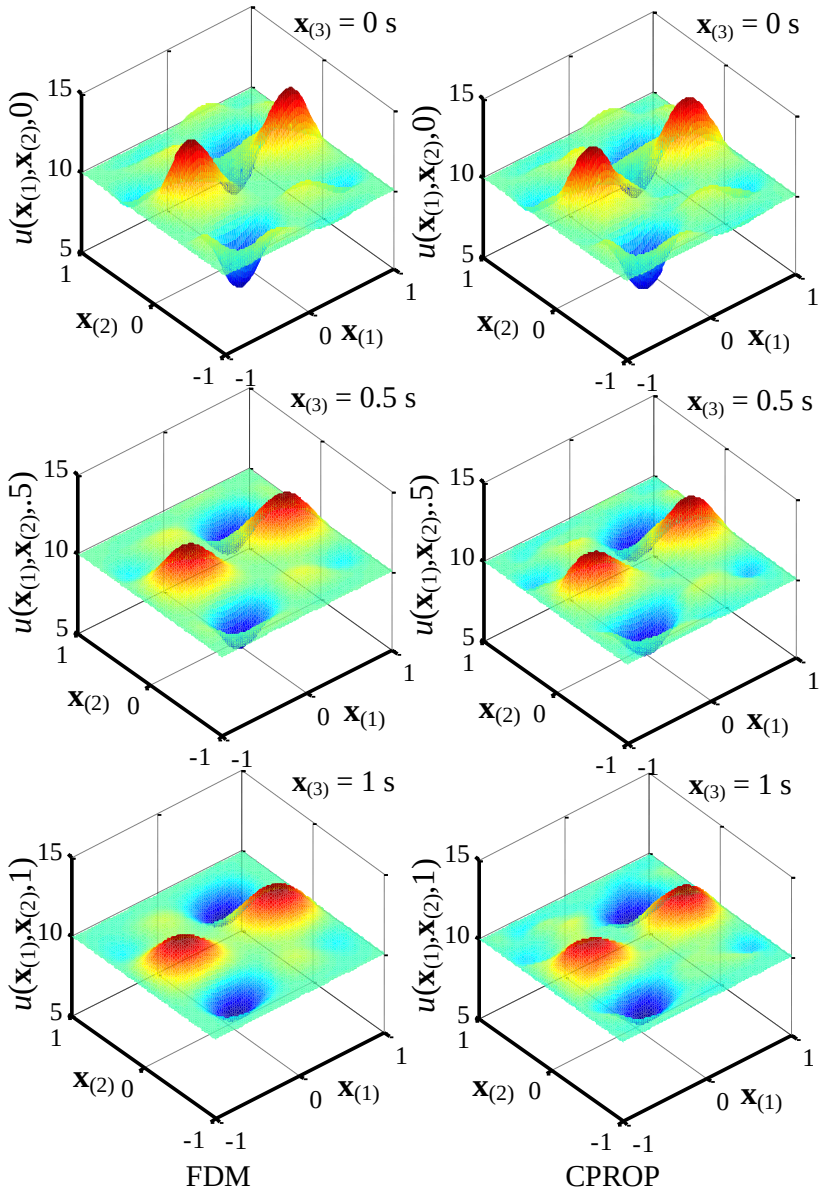


FIGURE 4.12: Solution of Boussinesq PDE (4.10) obtained by CPROP when $n = 0$ is compared to FDM solution at $\mathbf{x}_{(3)} = 0$ s, $\mathbf{x}_{(3)} = 0.5$ s, and $\mathbf{x}_{(3)} = 1$ s.

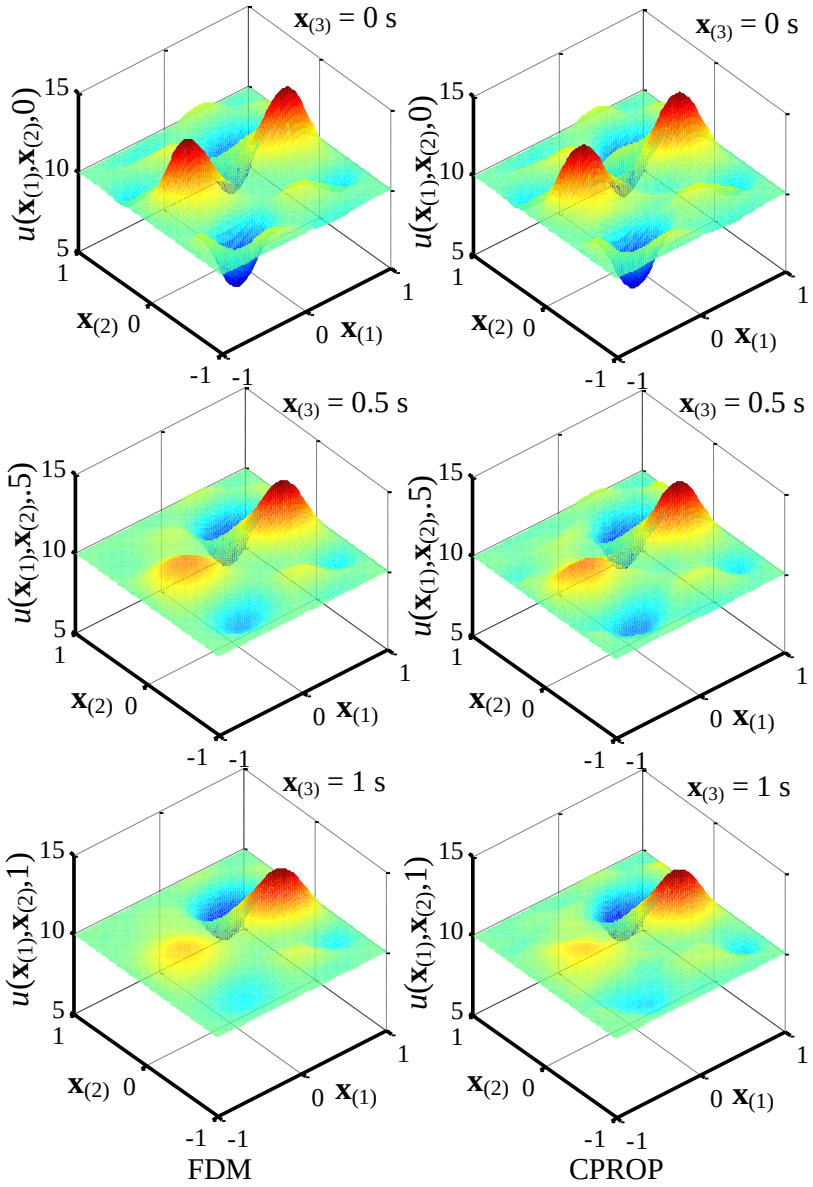


FIGURE 4.13: Adaptive solution of Boussinesq PDE (4.10) obtained by CPROP when $n = 1$ is compared to (non-adaptive) FDM solution at $\mathbf{x}_{(3)} = 0$ s, $\mathbf{x}_{(3)} = 0.5$ s, and $\mathbf{x}_{(3)} = 1$ s.

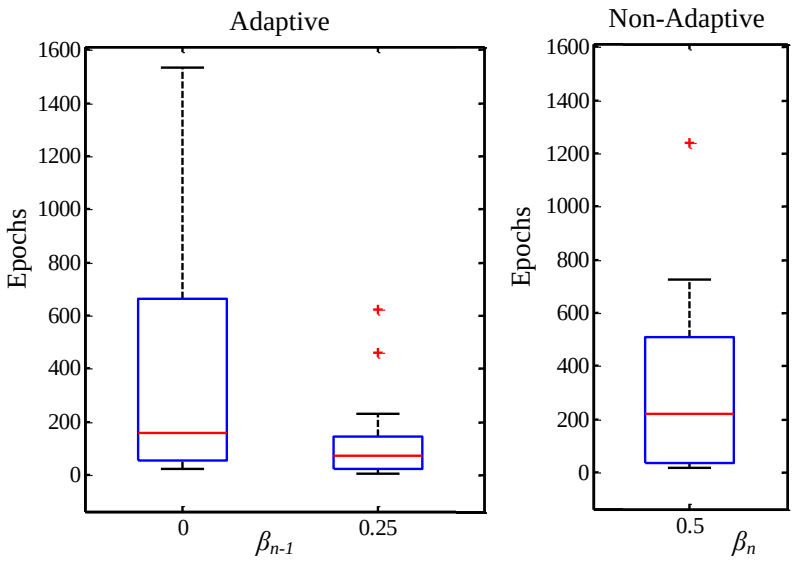


FIGURE 4.14: Box plots of number of CPROP training epochs needed to solve the Boussinesq PDE (4.10) adaptively and non-adaptively.

tive CPROP solution brings about a significant reduction in the number of training epochs required.

4.2 CINT Simulations and Results

The method presented in the previous section is applied to three IBVPs described in this section. The first consists of a 2D wave equation in a circular domain with a Dirichlet boundary condition. The second problem is a 2D wave equation solved over a square domain with a Neumann boundary condition. The wave equation is a linear hyperbolic PDE and was chosen for its wide use in areas ranging from acoustics [78] to electromagnetics [96]. The final problem is the 2D heat/diffusion equation on a semi-circular domain with a Dirichlet boundary condition. This equation is a parabolic type PDE, and has been used to model physical phenomena such as particle and thermal diffusion, and also arises in other areas such as financial mathematics. These PDEs and their domains were chosen because in each case a simple analytical solution is available, and, thus, can be used to compare and validate the results.

For each IBVP, comparisons were made between the CINT method and Matlab's FE-based PDE toolbox solver, including a comparison of accuracy and speed. The FE method was chosen to provide a baseline comparison, as it is frequently used in problems with complex geometries [15]. It has also been used extensively to solve the wave and heat/diffusion equations [4, 3, 2, 43, 111]. It is noted that both methods used Matlab's *ODE15s* ODE solver [94] to integrate the respective ODEs that arise in each method. As the speed of the algorithm is determined by the computational complexity and the allowable integration step size, the mean step size, Δt, observed in each method is also reported. As *ODE15s* uses an adaptive time step, the mean observed step size gives an approximation of the allowable step size for these problems. The error from the hyperbolic IBVPs is measured using the root mean square (RMS) error

$$RMS(t) = \sqrt{\frac{\sum_{m=1}^{M}[u(\mathbf{x}_m, t) - \hat{u}(\mathbf{x}_m, t)]^2}{M}}, \tag{4.16}$$

for M points in $\mathcal{I}$. Because for the parabolic IBVP, $u \to 0$ as $t \to \infty$, a more meaningful measure of solution accuracy is the relative error norm (REN)

$$REN(t) = \frac{\sqrt{\sum_{m=1}^{M}[u(\mathbf{x}_m, t) - \hat{u}(\mathbf{x}_m, t)]^2}}{\sqrt{\sum_{p=1}^{P} u^2(\mathbf{x}_p, t)}}. \tag{4.17}$$

The observed statistics of the errors from the IBVPS are shown in Table 4.1. The errors reported are the cumulative RMS errors for the IBVPs in Section 4.2.1 and 4.2.2 and the cumulative REN for the IBVP in Section 4.2.3. Mean Δt is the observed mean time step used in the integration, and Time is the computational time, in seconds, required for each problem. It can be seen that in both hyperbolic problems, the CINT method obtained the numerical solution significantly faster than the FE method. In the heat/diffusion equation, the FE method was found to be faster. In

Table 4.1: Observed statistics from the three IBVPs.

IBVP	Method	Error	Mean Δt	Time [s]
(4.18)	CINT	.0018	.006	6.9
(4.18)	FE	.0064	.003	49.2
(4.25)	CINT	.013	.0046	2.18
(4.25)	FE	.11	.001	38.65
(4.31)	CINT	.0018	.147	3.1
(4.31)	FE	.0034	.128	1.7

each IBVP studied the solution obtained from the CINT method was considerably more accurate than the solution provided by the FE method.

4.2.1 Wave Equation with a Dirichlet Boundary Condition in Two Dimensions

The results obtained by the CINT method for the two dimensional wave equation are given. Note that throughout this section $\mathbf{x} = [x, y]^T$. The IBVP was solved over the circular domain given by $\mathcal{I} = \{(x, y) \mid x^2 + y^2 \leqslant 1\}$, and over the time interval $t \in [0, 3]$. The wave equation is given by

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \tag{4.18}$$

where c is the wave speed, taken to be 2. The above PDE is subject to the Dirichlet boundary condition

$$u(x, y, t) = 0 \quad (x, y) \in \partial\mathcal{I}, \tag{4.19}$$

and initial conditions

$$u(x, y, 0) = J_0 \left(\lambda_4 \sqrt{x^2 + y^2} \right) \quad (x, y) \in \mathcal{I}, \tag{4.20}$$

$$\frac{\partial u}{\partial t}(x, y, 0) = 0 \quad (x, y) \in \mathcal{I}. \tag{4.21}$$

$J_0(\cdot)$ represents a Bessel function of the first kind,

$$J_0(r) = \sum_{m=0}^{\infty} \frac{(-1)^m}{(m!)^2} \left(\frac{x}{2} \right)^{2m}, \tag{4.22}$$

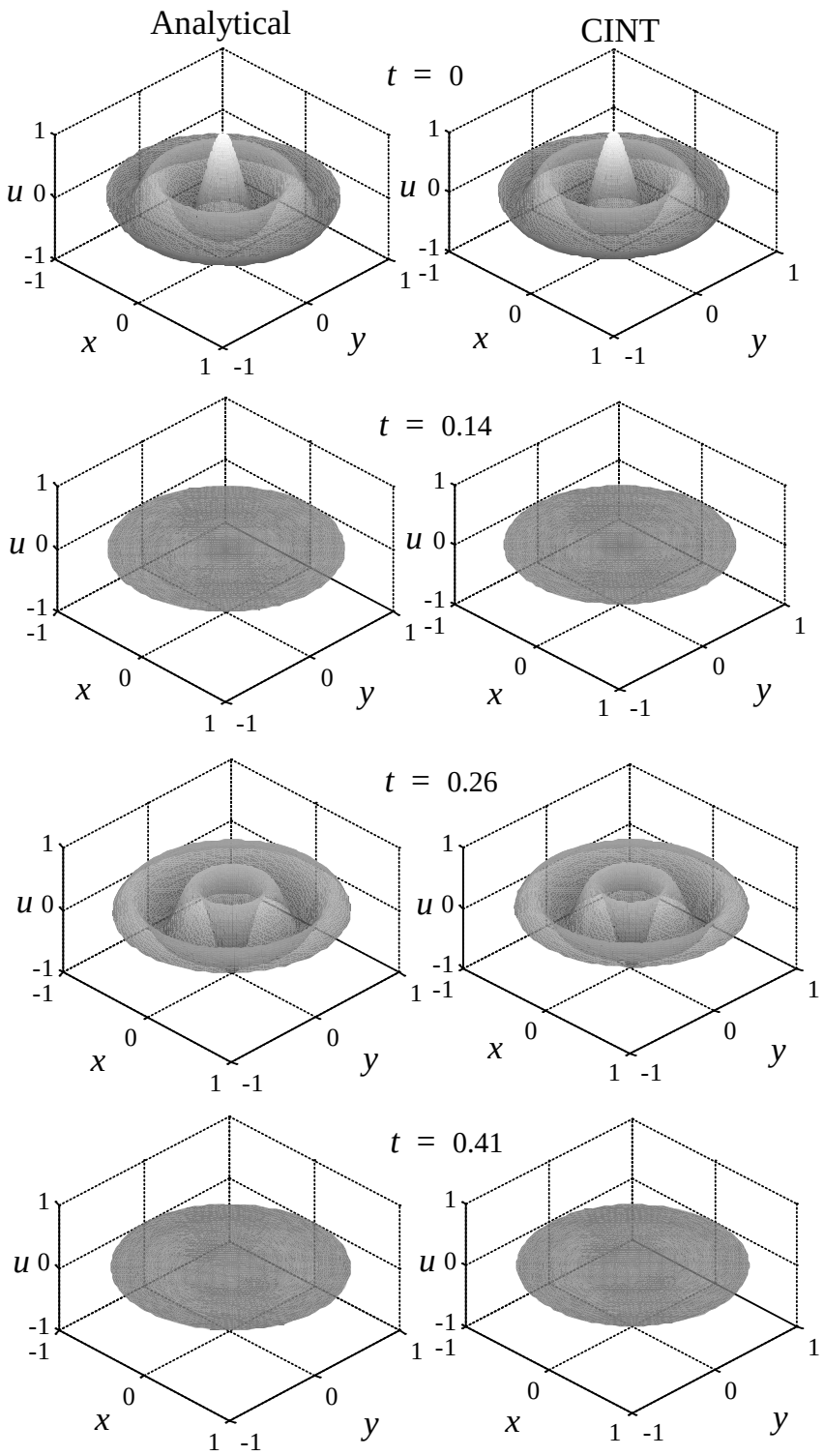


FIGURE 4.15: Analytical (left) and numerical (right) solutions to (4.18) at $t = 0$ to $t = .41$.

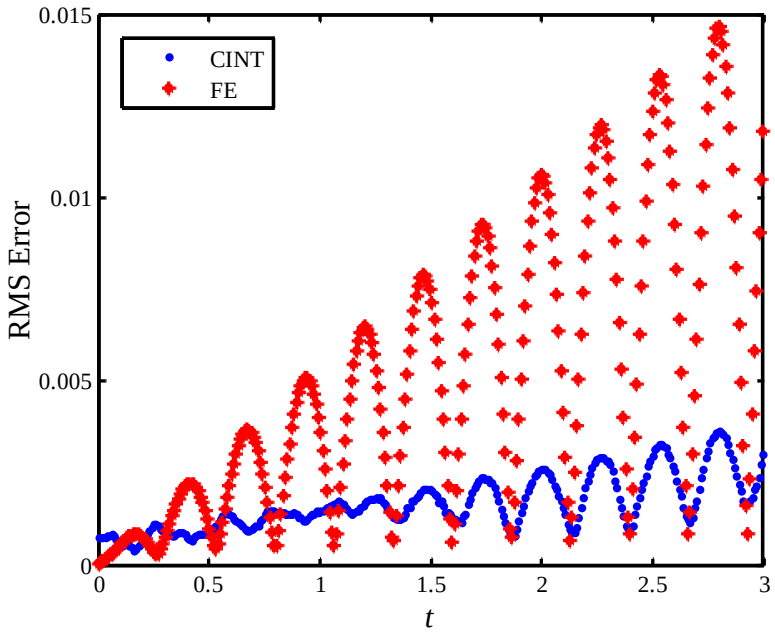


FIGURE 4.16: RMS error in the approximate solutions to (4.18) returned by the FE and CINT methods versus t.

and λ_4 represents the 4th zero of $J_0(\cdot)$. This IBVP has the analytical solution

$$u(x, y, t) = J_0\left(\lambda_4\sqrt{x^2 + y^2}\right)\cos(c\lambda_4 t). \tag{4.23}$$

The RBFs, $\{\tilde{\sigma}_1(\mathbf{x}), ..., \tilde{\sigma}_{\tilde{Q}}(\mathbf{x})\}$, that were used for this problem are given by (3.35) with $\gamma = 10$ and $\tilde{Q} = 40$. They were centered at 40 points uniformly distributed along the boundary of the domain. A polynomial basis was used for the transfer functions $\sigma_{j,m}(\mathbf{x})$,

$$\sigma_{j,m}(x, y) = x^{j-m}y^m, \tag{4.24}$$

where $j = 0, ..., 14$, $m = 0, ..., j$.

The analytical and numerical solutions to (4.18) are shown in Fig. 4.15. The RMS errors observed in the solutions obtained by the FE and CINT methods are shown in Fig. 4.16. It can be seen that the initial error is slightly larger in the solution returned from the CINT method, however, the error grows more slowly

than the observed error in the solution obtained by the FE method. Furthermore, the CINT method arrived at the numerical solution in approximately 6.9 seconds and Matlab's solver required approximately 49.2 seconds. Thus, the CINT method reduced the computation time by 85%. The cumulative RMS error for this problem was 0.0064 for the FE method and 0.0018 for the CINT method, an error reduction of approximately 70%.

4.2.2 Wave Equation with a Neumann Boundary Condition in Two Dimensions

The results of applying the CINT method to the 2D wave equation over a square domain with a Neumann boundary condition are given in this section. The PDE was solved over the domain $\mathcal{I} = \{(x, y) \mid (x, y) \in [-1, 1] \times [-1, 1]\}$, and time interval $t \in [0, 3]$. The wave equation is again given by

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \tag{4.25}$$

and is subject to the boundary condition

$$\nabla u(x, y, t) \cdot \hat{\mathbf{n}} = 0 \quad (x, y) \in \partial\mathcal{I}, \tag{4.26}$$

where $\hat{\mathbf{n}}$ is the outward unit normal vector. The initial conditions are given by

$$u(x, y, 0) = \cos(2\pi x) \cos(3\pi y) \quad (x, y) \in \mathcal{I}, \tag{4.27}$$

$$\frac{\partial u}{\partial t}(x, y, 0) = 0 \quad (x, y) \in \mathcal{I}. \tag{4.28}$$

This IBVP has the analytical solution

$$u(x, y, t) = \cos(2\pi x) \cos(3\pi y) \cos\left(c\pi\sqrt{2^2 + 3^2}t \right). \tag{4.29}$$

The RBFs used to solve this problem are given by (3.36), with $\gamma = 90$. The RBFs were centered at points distributed uniformly along the boundary, with 70 along each

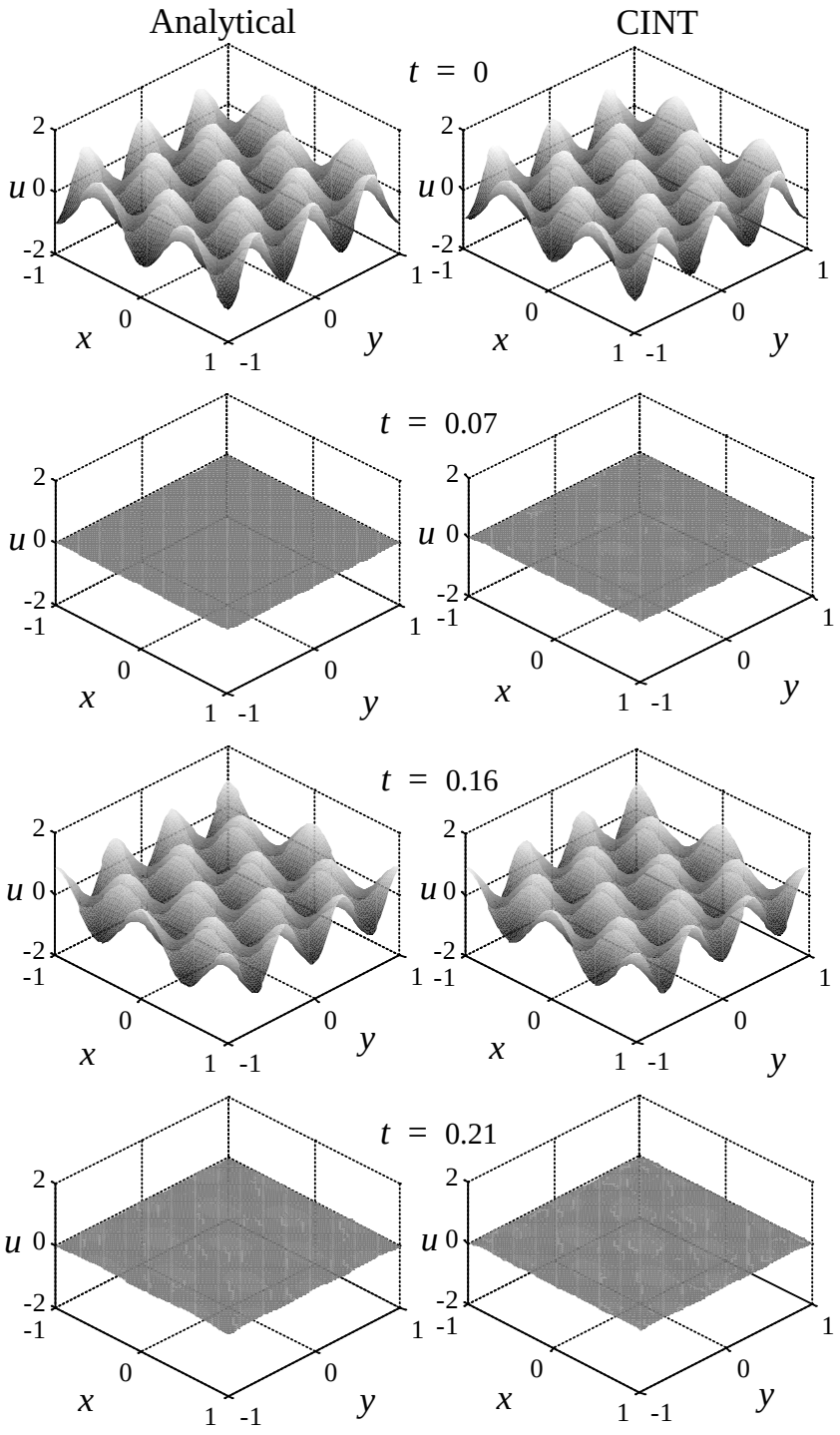


FIGURE 4.17: Analytical (left) and numerical (right) solutions to (4.25) at times $t = 0$, $t = .07$, $t = .16$, and $t = .21$.

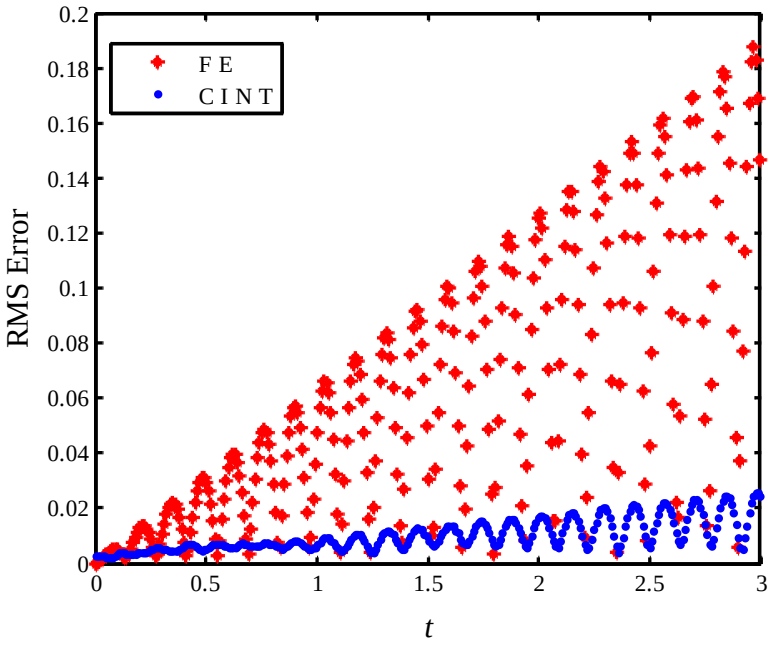


FIGURE 4.18: RMS error in the solutions to (4.25) returned by the FE and CINT methods versus t.

of the four sides of the square domain. As in the previous problem, a polynomial basis was used for the transfer functions, $\sigma_j(\mathbf{x})$, given by

$$\sigma_j(\mathbf{x}) = x^{m_j} y^{n_j}, \tag{4.30}$$

where $j = 0, ..., 14^2$, and $m_j, n_j \in \{0, ..., 14\}$. The analytical and numerical solutions to (4.25) are shown in Fig. 4.17. The left column shows the analytical solution; the right column shows the numerical solution obtained by the CINT method.

As in the previous IBVP, the RMS error was computed at various time steps and plotted in Fig. 4.18. It can be seen from this figure that the initial error is initially slightly larger in the solution obtained from the CINT method, however, grows significantly slower than the error measured in the solution obtained from the FE method. For this problem, the CINT method required approximately 2.18 seconds to obtain a solution, and Matlab's FE-based solver took approximately 38.65 seconds. Therefore, the CINT method reduced the computation time by 94%. The

cumulative RMS error observed in the solution obtained by the FE method was 0.11, wherease the solution obtianed with the CINT method had a cumulative RMS error of 0.013, an error reduction of approximately 88%.

4.2.3 Heat/Diffusion Equation in Two Dimensions

The final IBVP presented in this paper is given by the heat/diffusion equation in two spatial dimensions. This problem was solved in the domain given by the semicircle $\mathcal{I} = \{(x, y) \mid x \in [-1, 1], y \in \left[0, \sqrt{(1 - x^2)}\right]\}$, in the time interval $t \in [0, 5]$. The parabolic PDE is, then, given by

$$\frac{\partial u}{\partial t} = k \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \quad \mathbf{x} \in \mathcal{I}, \tag{4.31}$$

where k is the diffusivity, taken to have the constant value of .002. The above PDE is subject to the boundary condition

$$u(\mathbf{x}, t) = 0 \quad \mathbf{x} \in \partial\mathcal{I}, \tag{4.32}$$

and initial condition

$$u(\mathbf{x}, t_0) = J_3 \left(\lambda_3 \sqrt{x^2 + y^2} \right) \sin[3 \arctan(y/x)]. \tag{4.33}$$

$J_3(\cdot)$ in the above equation is a Bessel function of the first kind

$$J_3(r) = \sum_{m=0}^{\infty} \frac{(-1)^m}{m!(m+3)!} \left(\frac{x}{2} \right)^{2m+3}, \tag{4.34}$$

and λ_3 is the 3rd zero of the above Bessel function. This problem has the analytical solution

$$u(\mathbf{x}, t) = J_3 \left(\lambda_3 \sqrt{x^2 + y^2} \right) \sin[3 \arctan(y/x)] e^{-k\lambda_3^2 t}. \tag{4.35}$$

As in the problem presented in Section 4.2.1, the RBFs used in the approximate solution are given by (3.35), with $\gamma = 20$. Sixty RBFs centered at points distributed

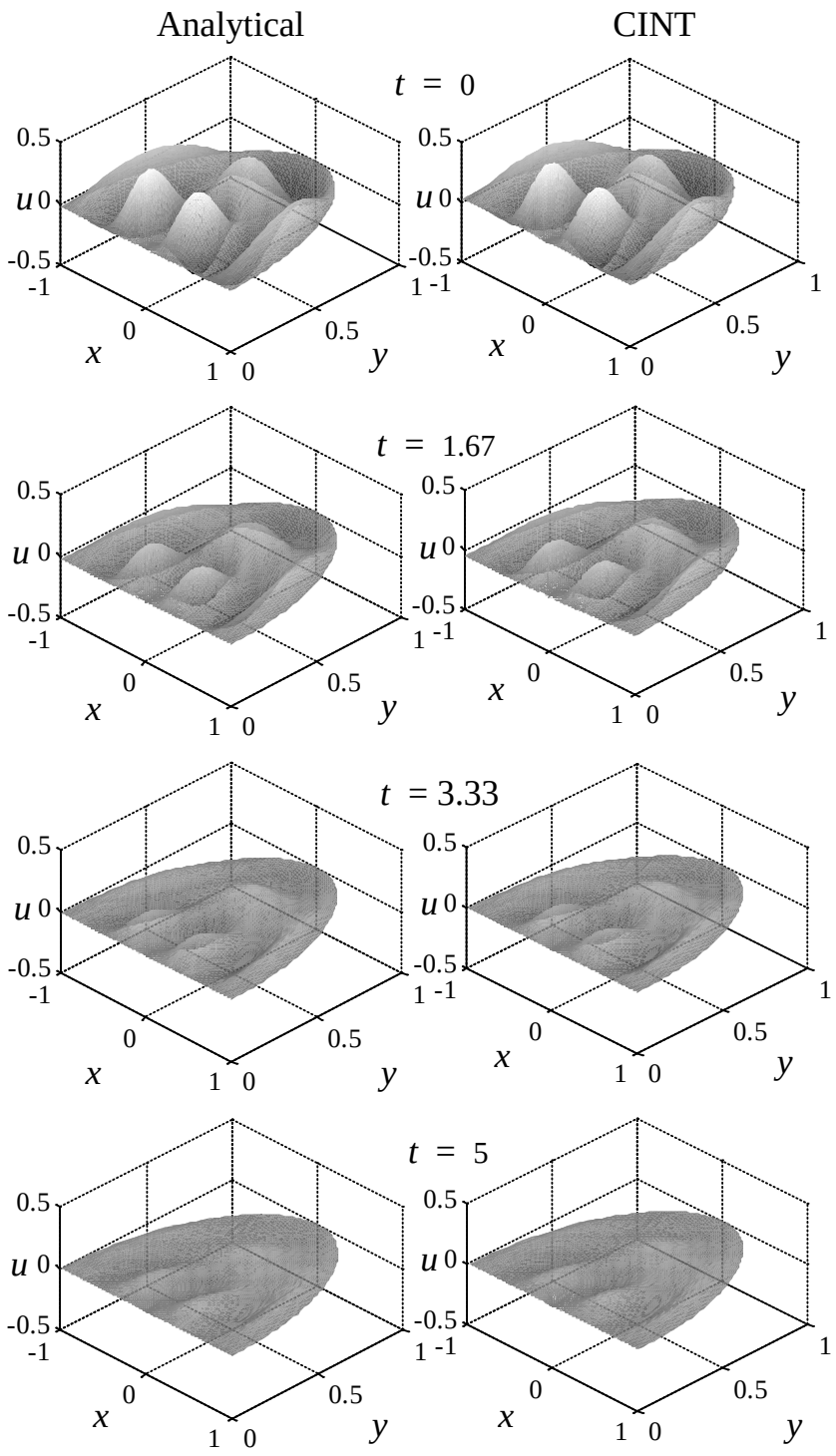


FIGURE 4.19: Analytical (left) and numerical (right) solutions to (4.31) at times $t = 0$, $t = 1.67$, $t = 3.33$, and $t = 5$.

uniformly over the boundary of the domain were used. A truncated Fourier basis was used for the transfer functions $\sigma_j(\mathbf{x})$:

$$\sigma_j(\mathbf{x}) \in \{1, \sin(\pi x), \cos(\pi x), ..., \sin(5\pi x), \cos(5\pi x)\} \otimes$$
$$\{1, \sin(\pi y), \cos(\pi y), ..., \sin(5\pi y), \cos(5\pi y)\}, \tag{4.36}$$

where $\otimes$ denotes the tensor product.

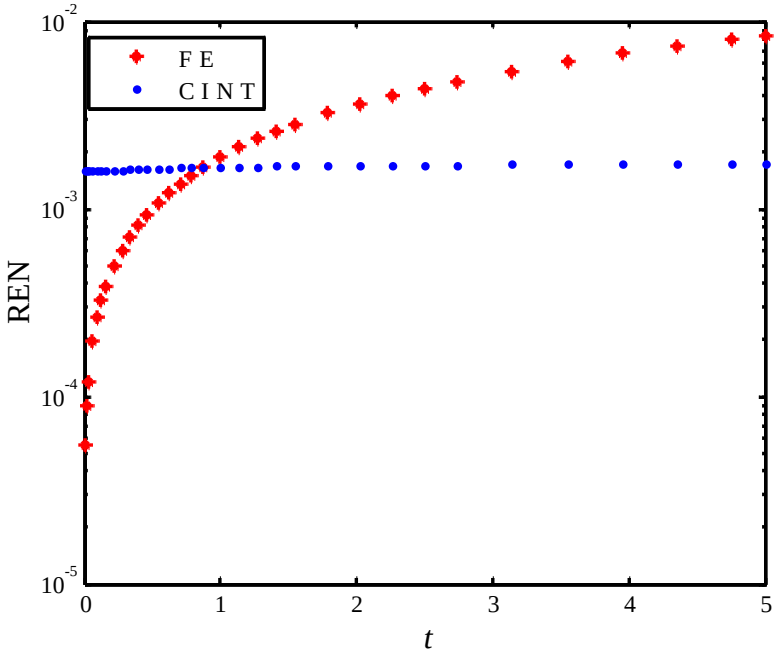


FIGURE 4.20: REN observed in the FE and CINT numerical solutions to (4.31).

The analytical and numerical solutions to (4.31) are shown in Fig. 4.19 at times $t = 0, 1.67, 3.33, 5$. The computational time was approximately 0.6 seconds for the FE method and 1.7 seconds for the CINT method. The REN observed in the solution returned by the FE and CINT methods are shown in Fig. 4.20. This plot shows that, although initially more accurate, the error in the solution from the FE method grows significantly faster than the error measured in the solution from the CINT method, and quickly becomes less accurate. An analysis of the error, or rate of convergence of the CINT method, is provided in the following subsection.

4.2.4 Error Analysis

The rate at which the CINT method is able to solve PDEs is determined by the number of basis functions required to satisfy a user defined error tolerance. Explicitly, assuming that $\mathbf{M}^+$ has been pre-computed and stored, each step within the ODE solver requires $O(NQ)$ computations. It has been shown that if $\mathbf{u}$ is analytic, then classical spectral methods converge exponentially in Q [14],

$$||u(\mathbf{x},t) - \hat{u}(\mathbf{x},t)|| \leqslant c_1 e^{-c_2 Q}, \tag{4.37}$$

where c_1 and c_2 are positive constants. Exponential convergence was also observed in the CINT method. Figure 4.21 shows the RMS error versus the highest degree of polynomial used to solve the IBVP given in section 4.2.2 (or $\sqrt{Q_S}$ as the PDE is solved in two dimensions). It was found that $RMS \approx 692 \exp(-0.85\sqrt{Q_S})$, and is the solid line plotted in Fig. 4.21.

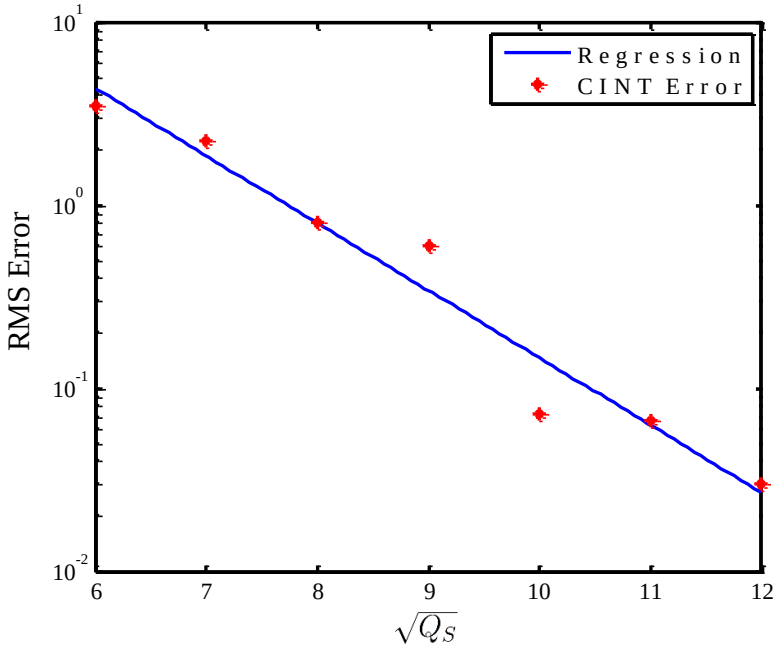


FIGURE 4.21: Exponential convergence observed in the CINT method. This plot shows the RMS error versus the power of polynomial used in the approximate solution (3.42). The observed errors are indicated by stars (*), and the exponential regression is shown by the solid line.

5

Distributed Optimal Control (DOC)

This chapter considers the problem of computing optimal state and control trajectories for a multiscale dynamical system comprised of many interacting dynamical systems, or agents. Optimality conditions are derived and solved using the CINT method to obtain optimal control.

5.1 Distributed Optimal Control

Many complex systems ranging from renewable resources [89] to very large scale robotic (VLRS) systems [84] can be described as multiscale dynamical systems comprised of many interactive agents. In recent years, significant progress has been made in formation control and stability analysis of teams of robots, or swarms, in which the mutual goal of the agents is to maintain a desired configuration, such as a triangle or a star formation, or a desired behavior, such as translating as a group (schooling), or maintaining the center of mass of the group stationary (flocking) [7, 25, 35, 29, 61, 84]. While this literature has successfully illustrated that the behavior of large networks of interacting agents can be conveniently described and

controlled by density functions, it has yet to provide an approach for controlling the agents such that their overall performance is optimized.

Recently, a coarse-grained optimal control approach for large, multiscale dynamical systems, referred to as distributed optimal control (DOC) was proposed, that enables the optimization of density functions, and/or their moments, subject to the agents' dynamic constraints [34]. The DOC approach in [34] is applicable to multiscale dynamical systems comprised of many agents or processes that, on small spatial and time scales, are each described by a small set of ordinary differential equations (ODEs), referred to as the microscopic or *detailed* equations. On larger spatial and temporal scales, the agents' dynamics and interactions are assumed to give rise to macroscopic coherent behaviors, or *coarse dynamics*, described by partial differential equations (PDEs). This chapter extends the capabilities of the DOC approach proposed in [34] for deterministic agent dynamics, to agent dynamics that are governed by stochastic differential equations (SDEs).

In recent years, the optimal control of stochastic differential equations (SDEs) has gained increasing attention. Considerable research efforts have focused on the optimal control and estimation of SDEs driven by non-Gaussian processes, such as Brownian motion combined with Poisson processes, and various other stochastic processes [99, 100, 76]. The approach in [99, 100, 76] views the microscopic agent state as a random vector, and derives an SDE dynamic equation that involves the evolution of the statistics of the microscopic vector function, and may be integrated using stochastic integrals. Then, the performance of multiple agents can be expressed as an integral function of multiple, corresponding vector fields to be optimized subject to a set of SDEs. However, solutions can only be obtained for relatively few and highly idealized cases in which finite-dimensional, local approximations can be constructed, for example, via moment closure [99, 100]. Therefore, while optimal control of SDEs has been shown useful to selected applications in population biology and finance

[99, 100, 76], it is yet to be successfully applied to multiscale systems in which the coarse dynamics do not obey these idealized conditions, and are instead dictated by realistic constraints (e.g., vehicle dynamics) and objectives (e.g., minimizing energy consumption, or maximizing coverage).

The GRG-DOC methodology presented in this chapter relies on identifying a consistency relationship between the microscopic agent dynamics and a macroscopic description, such as the time-varying probability density function (PDF) of the agents' state. Unlike Nash Certainty Equivalence (NCE), or Mean Field, methods, in which the (weak) couplings between agents are produced by the averaging of the microscopic agent dynamics and costs, in the DOC approach the couplings need not be weak, and may arise as a result of cooperative objectives expressed by the macroscopic cost function. Therefore, the cost function can represent objectives of a far more general form than NCE, and admit (optimal) solutions that entail strong couplings between the agent dynamics and control laws. Also, unlike prioritized and path-coordination methods [106, 58], the proposed DOC approach does not rely on decoupling the agents' dynamics, or on specifying the agents' distribution *a priori*. Instead, DOC optimizes the macroscopic behavior of the system subject to coupled microscopic agent dynamics, and relies on the existence of an accurate macroscopic evolution equation and an associated restriction operator that characterize the multiscale system to reduce the computational complexity of the optimal control problem. As a result, the computation required is far reduced compared to classical optimal control, and realizations of the trajectories of all agents over large spatial and time scales are calculated simultaneously without sacrificing optimality or completeness.

5.2 Problem Formulation

This chapter considers the problem of computing optimal state and control trajectories for a multiscale dynamical system comprised of N interacting dynamical systems,

or agents. The dynamics of each agent on the microscopic scale can be described by a small system of the SDEs,

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t] + \mathbf{G}\mathbf{w}(t), \quad \mathbf{x}(t_0) = \mathbf{x}_0, \tag{5.1}$$

where $\mathbf{x}(t) \in \mathcal{X} \subset \mathbb{R}^n$ is the microscopic state and $\mathbf{u} = c[\mathbf{x}(t), t] \in \mathcal{U} \subset \mathbb{R}^m$ is the microscopic control law, which is assumed to be a function of the state. The microscopic dynamics are influenced by additive Gaussian noise, where the disturbance, $\mathbf{w} \in \mathbb{R}^n$, is a vector of independent and identically distributed random variables from a standard Gaussian process, and $\mathbf{G}$ is a time-invariant matrix. A standard Gaussian process is used here for simplicity, but this approach is applicable to any diffusion process. It is assumed that the microscopic state, $\mathbf{x}$, of every agent is fully observable and error free.

On large spatial and temporal scales, the agents can be represented by a macroscopic state, denoted by $\mathbf{X} \in \mathbb{R}^\ell$, $\ell << n$, by means of a restriction operator. Depending on the macroscopic system performance to be optimized, the restriction operator may consist of the agent distribution and/or of its lower-order moments, such that $X(t) = \wp[\mathbf{x}(t), t]$ [51]. In this chapter, the system restriction operator $\wp$ is assumed to be a time-varying probability density function (PDF), $\wp : \mathcal{X} \times \mathbb{R} \to \mathbb{R}$, such that the probability that the state of the i^{th} agent has a value $\mathbf{x} \in B \subset \mathcal{X}$ is given by,

$$P[\mathbf{x}(t) \in B] = \int_B \wp[\mathbf{x}(t), t] d\mathbf{x} \tag{5.2}$$

Then, the agent PDF, $\wp$, is a non-negative probability function that must satisfy the normalization condition,

$$\int_{\mathcal{X}} \wp[\mathbf{x}(t), t] d\mathbf{x} = 1 \tag{5.3}$$

In many complex systems, such as autonomous vehicles and sensor networks, the performance to be optimized can be defined as an integral cost function of the

macroscopic state $\wp$ and the microscopic control $\mathbf{u}$,

$$J = \int_{\mathcal{X}} \phi\left[\wp(\mathbf{x}, t_t), t_f\right] d\mathbf{x} + \int_t \int_{\mathcal{X}} \mathscr{L}[\wp(\mathbf{x}, t), \mathbf{u}(t), t] d\mathbf{x} dt, \tag{5.4}$$

where $\mathscr{L}$ is the Lagrangian function of the DOC problem. The multi-agent trajectory optimization problem considered in this chapter seeks to determine the optimal trajectories for the macroscopic state X^* and microscopic control $\mathbf{u}^*$ that minimizes the cost function (5.4), subject to the dynamic constraint (5.1) and the equality constraint (5.3).

Assuming that the agents exist only in the state space $\mathcal{X}$,

$$\wp[\mathbf{x}(t) \notin \mathcal{X}, t] = 0, \quad \forall \in (t_0, t_f] \tag{5.5}$$

and that no agents are created or destroyed, the evolution of the agent PDF, can be shown to be governed by the so-called advection-diffusion equation. The advection-diffusion equation is a parabolic PDE that describes the motion of a conserved scalar quantity, such as a PDF, as it is advected by a known velocity field and undergoes a diffusion process [13]. Since the agent distribution, $\wp$, is advected by a known velocity field $\mathbf{v} = \dot{\mathbf{x}} \in \mathbb{R}^n$, given by the detailed equation (5.1), and diffused by the additive Gaussian noise, the time-rate of change of $\wp$ can be defined as the sum of the negative divergence of the advection vector ($\wp\mathbf{v}$) and the divergence of diffusion vector ($\mathbf{G}\mathbf{G}^T\nabla\wp$) [68]. Then, from the advection-diffusion equation, the agent PDF is governed by,

$$\frac{\partial \wp}{\partial t} = -\nabla \cdot \{\wp(\mathbf{x}, t)\mathbf{f}[\mathbf{x}, \mathbf{u}(t), t]\} + \nu\nabla^2\wp(\mathbf{x}, t) \tag{5.6}$$

where the ∇ denotes a row vector of partial derivatives with respect to the elements of $\mathbf{x}$, and the diffusion coefficient is $\nu = \mathbf{G}\mathbf{G}^T$.

This chapter presents a GRG method for computing the optimal trajectories of the macroscopic state X^* and the microscopic control $\mathbf{u}^*$ that optimize J over the

time interval $(t_0, t_f]$. The optimization of J is subject to the macroscopic dynamics (5.6), the normalization condition (5.3), and the state space constraint (5.5). and the initial and boundary conditions. Additionally, since the agents are assumed to exist in $\mathcal{X}$ at all times, their initial PDF, g_0 is typically given. Therefore, the optimization of J is also subject to the initial and boundary conditions,

$$\wp(\mathbf{x}, t_0) = g_0(\mathbf{x}) \tag{5.7}$$

$$\{\nabla \wp(\mathbf{x}, t)\} \cdot \hat{\mathbf{n}} = 0, \quad \forall \in (t_0, t_f] \tag{5.8}$$

that require all agents to remain in $\mathcal{X}$ at all times, where $\hat{\mathbf{n}}$ is a vector normal to $\partial\mathcal{X}$ of unit length. The following section presents an indirect solution method based on a GRG approach for solving PDE-constrained optimization problems.

5.3 Methodology

An indirect GRG solution method is presented in this section for computing the optimal macroscopic state and microscopic control trajectories for the DOC problem in (5.1)-(5.6). By this approach, a Lagrange multiplier, $\lambda(\mathbf{x}, t)$, is used to adjoin the dynamic and equality constraints, (5.5)-(5.8), (5.3), to the integral cost function (5.4), obtaining the augmented integral cost function,

$$\hat{J}(\wp, \mathbf{u}, \lambda) = \int_{\mathcal{X}} \phi\left\{\wp(\mathbf{x}, t_f), t_f\right\} d\mathbf{x} + \int_t \int_{\mathcal{X}} \Bigg\{ \mathscr{L}\left[\wp(\mathbf{x}, t), \mathbf{u}(t)\right] \tag{5.9}$$

$$+\lambda(\mathbf{x}, t)\Big[\frac{\partial \wp(\mathbf{x}, t)}{\partial t} + \nabla \cdot \left[\wp(\mathbf{x}, t)\mathbf{f}(\mathbf{x}, \mathbf{u}, t)\right] - \nu \nabla^2 \wp(\mathbf{x}, t)\Big]\Bigg\} d\mathbf{x} dt.$$

The necessary conditions for optimality for this augmented cost function are derived in Section 5.3.1 using the calculus of variations.

To have a closed form representation of the control for all $\mathbf{x}$, every element of the control vector, u_j, is parameterized as the sum of m linearly-independent basis

functions $\phi_1(\cdot), \ldots, \phi_m(\cdot)$, such that

$$u_j = \sum_k \phi_k(\mathbf{x})\alpha_{j,k}(t), \quad \text{for } j = 1, \ldots, m. \tag{5.10}$$

Then, the goal of the DOC problem is to obtain the parameters, $\alpha^*_{j,k}(t)$, that minimize the cost function (5.4), subject to the aforementioned constraints. As shown in Section 5.3.2, since the macroscopic state, $\wp$, and the Lagrangian multiplier, λ, can be found explicitly as a function of $\mathbf{u}$, a generalized reduced gradient (GRG) method [105] can be used to determine the optimal parameters of the control law (5.10).

5.3.1 Optimality Conditions

The optimality conditions for the optimal control problem presented in Section 5.2 are derived here using calculus of variations. Let $\boldsymbol{\mu} = [\mathbf{u}^T, \wp, \lambda]^T$ denote a vector of variables for the DOC problem, where function arguments are omitted hereon for brevity. The necessary condition for optimality is,

$$\nabla \hat{J}(\mathbf{u}, \wp, \lambda) = \lim_{\epsilon \to 0} \frac{\hat{J}(\boldsymbol{\mu} + \epsilon\delta\boldsymbol{\mu}) - \hat{J}(\boldsymbol{\mu})}{\epsilon} = 0, \tag{5.11}$$

where $\nabla \hat{J}$ is the gradient of $\hat{J}$ with respect to the variables, $\mathbf{u}$, $\wp$, λ, and the vector $\epsilon\ \delta\boldsymbol{\mu} = \epsilon\ [\delta\mathbf{u}^T, \delta\wp, \delta\lambda]^T$ contains the variations of the DOC variables.

The variation in the PDF, $\wp \to \wp + \epsilon\wp$, results in the condition,

$$\lim_{\epsilon \to 0} \frac{\hat{J}(\mathbf{u}, \wp + \epsilon\delta\wp, \lambda) - \hat{J}(\mathbf{u}, \wp, \lambda)}{\epsilon} = \int_{\mathcal{X}} \frac{\partial \phi}{\partial \wp}\Big|_{t_f} \delta\wp d\mathbf{x} \tag{5.12}$$

$$+ \int_t \int_{\mathcal{X}} \frac{\partial \mathscr{L}}{\partial \wp}\delta\wp + \lambda \left[\frac{\partial \delta\wp}{\partial t} + \nabla \cdot (\delta\wp\ \mathbf{f}) - \nabla^2 \delta\wp \right] d\mathbf{x}dt = 0.$$

which provides the weak formulation of the DOC optimality conditions. The fundamental theorem of variational calculus (ftvc) is used to arrive at the strong formulation of the DOC optimality conditions. From the ftvc, and integration by parts, the

partial derivatives acting on the variations are

$$\int_t \int_{\mathcal{X}} \lambda \frac{\partial \delta\wp}{\partial t} d\mathbf{x}dt = \tag{5.13}$$
$$\int_{\mathcal{X}} \lambda \delta\wp d\mathbf{x}\Big|_{t_0}^{t_f} - \int_t \int_{\mathcal{X}} \frac{\partial \lambda}{\partial t} \delta\wp d\mathbf{x}dt,$$

$$\int_t \int_{\mathcal{X}} \lambda \nabla \cdot (\delta\wp \, \mathbf{f}) \, d\mathbf{x}dt = \tag{5.14}$$
$$\int_t \int_{\partial\mathcal{X}} \lambda (\mathbf{f} \cdot \hat{\mathbf{n}}) \delta\wp d\mathbf{x}dt - \int_t \int_{\mathcal{X}} \nabla\lambda \cdot \mathbf{f} \delta\wp d\mathbf{x}dt,$$

$$\int_t \int_{\mathcal{X}} \nu\lambda \nabla^2 \delta\wp d\mathbf{x}dt = \tag{5.15}$$
$$\int_t \int_{\partial\mathcal{X}} \nu\lambda (\nabla\delta\wp \cdot \hat{\mathbf{n}}) \delta\wp d\mathbf{x}dt - \int_t \int_{\mathcal{X}} \nu \nabla\lambda \cdot \nabla\delta\wp d\mathbf{x}dt =$$
$$\int_t \int_{\partial\mathcal{X}} \nu\lambda (\nabla\delta\wp \cdot \hat{\mathbf{n}}) \delta\wp d\mathbf{x}dt - \int_t \int_{\partial\mathcal{X}} \nu \nabla\lambda \cdot \hat{\mathbf{n}} \delta\wp d\mathbf{x}dt +$$
$$\int_t \int_{\mathcal{X}} \nu \nabla^2 \lambda \delta\wp d\mathbf{x}dt.$$

Because an initial condition for $\wp$ is given at t_0, as shown in (5.7), the initial variation in the PDF is $\delta\wp\big|_{t_0} = 0$, and (5.13) simplifies to

$$\int_t \int_{\mathcal{X}} \lambda \frac{\partial \delta\wp}{\partial t} d\mathbf{x}dt = \tag{5.16}$$
$$\int_{\mathcal{X}} \lambda \delta\wp d\mathbf{x}\Big|_{t_f} - \int_t \int_{\mathcal{X}} \frac{\partial \lambda}{\partial t} \delta\wp d\mathbf{x}dt.$$

The boundary condition (5.8) implies that (5.15) simplifies to

$$\int_t \int_{\mathcal{X}} \nu\lambda \nabla^2 \delta\wp d\mathbf{x}dt = \tag{5.17}$$
$$- \int_t \int_{\partial\mathcal{X}} \nu \nabla\lambda \cdot \hat{\mathbf{n}} \delta\wp d\mathbf{x}dt + \int_t \int_{\mathcal{X}} \nu \nabla^2 \lambda \delta\wp d\mathbf{x}dt.$$

Then, by substituting the results in (5.14), (5.16), and (5.17) into (5.12), and grouping like terms, the variation in (5.12) can be written as

$$0 = \int_{\mathcal{X}} \left(\frac{\partial \phi}{\partial \wp} + \lambda \right) \delta \wp \Big|_{t_f} d\mathbf{x} + \tag{5.18}$$

$$\int_t \int_{\partial \mathcal{X}} \left(\lambda (\mathbf{f} \cdot \hat{\mathbf{n}}) + \nu \nabla \lambda \cdot \hat{\mathbf{n}} \right) \delta \wp d\mathbf{x} dt +$$

$$\int_t \int_{\mathcal{X}} \left(\frac{\partial \mathscr{L}}{\partial \wp} - \frac{\partial \lambda}{\partial t} - \nabla \lambda \cdot \mathbf{f} - \nu \nabla^2 \lambda \right) \delta \wp d\mathbf{x} dt.$$

By the fundamental theorem of variational calculus (ftvc), the variation in (5.18) can be written as the adjoint PDE:

$$\frac{\partial \lambda}{\partial t} = \frac{\partial \mathscr{L}}{\partial \wp} - \nabla \lambda \cdot \mathbf{f} - \nu \nabla^2 \lambda \tag{5.19}$$

$$\text{SJT: } \lambda(\mathbf{x}, t_f) = -\frac{\partial \phi}{\partial \wp}\Big|_{t_f} \quad \mathbf{x} \in \mathcal{X},$$

$$\lambda(\mathbf{f} \cdot \hat{\mathbf{n}}) + \nu(\nabla \lambda) \cdot \hat{\mathbf{n}} = 0 \quad \mathbf{x} \in \partial \mathcal{X}$$

The variation in the control law, $\mathbf{u} \rightarrow \mathbf{u} + \epsilon \delta \mathbf{u}$,

$$\lim_{\epsilon \rightarrow 0} \frac{\hat{J}(\mathbf{u} + \epsilon \delta \mathbf{u}) - \hat{J}(\mathbf{u})}{\epsilon} = \tag{5.20}$$

$$\int_t \int_{\mathcal{X}} \frac{\partial \mathscr{L}}{\partial \mathbf{u}} + \lambda \left[\nabla \cdot \left(\wp \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \delta \mathbf{u} \right) \right] d\mathbf{x} dt =$$

$$\int_t \int_{\mathcal{X}} \frac{\partial \mathscr{L}}{\partial \mathbf{u}} - \nabla \lambda \cdot \left(\wp \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right) \delta \mathbf{u} d\mathbf{x} dt +$$

$$\int_t \int_{\partial \mathcal{X}} \lambda \left(\wp \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \cdot \hat{\mathbf{n}} \right) \delta \mathbf{u} d\mathbf{x} dt.$$

must equal zero for optimality, by the ftvc, i.e.:

$$0 = \frac{\partial \mathscr{L}}{\partial \mathbf{u}} - \nabla \lambda \cdot \left(\wp \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right). \tag{5.21}$$

Finally, the variation in the Lagrange multiplier, $\lambda \rightarrow \lambda + \epsilon\delta\lambda$, leads to the macroscopic state equation. Thus, the DOC optimality conditions are given by the set of PDEs:

$$\frac{\partial \wp}{\partial t} = -\nabla \cdot (\wp \mathbf{f}) + \nu \nabla^2 \wp \tag{5.22}$$

$$\text{SJT: } \wp(\mathbf{x}, t_0) = p(\mathbf{x}) \quad \mathbf{x} \in \mathcal{X},$$

$$\nabla \wp \cdot \hat{\mathbf{n}} = 0 \quad \mathbf{x} \in \partial\mathcal{X}$$

$$\frac{\partial \lambda}{\partial t} = \frac{\partial \mathscr{L}}{\partial \wp} - \nabla\lambda \cdot \mathbf{f} - \nu \nabla^2 \lambda \tag{5.23}$$

$$\text{SJT: } \lambda(\mathbf{x}, t_f) = -\frac{\partial \phi}{\partial \wp}\Big|_{t_f} \quad \mathbf{x} \in \mathcal{X},$$

$$\nabla\lambda \cdot \hat{\mathbf{n}} = 0 \quad \mathbf{x} \in \partial\mathcal{X}$$

$$0 = \frac{\partial \mathscr{L}}{\partial \mathbf{u}} - \nabla\lambda \cdot \left(\wp \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right). \tag{5.24}$$

The macroscopic state (5.22) and adjoint (5.23) equations are parabolic PDEs. The control equation (5.24) is an algebraic equation relating the optimal $\mathbf{u}$ to $\wp$ and λ. If (5.22)-(5.24) are satisfied, then the resulting $\wp$ and $\mathbf{u}$ are the optimal control and resulting agent distribution for the macroscopic control problem. To obtain the sufficient conditions for optimality, the second-order variations of $\hat{J}$ may be tested to verify that these values in fact are at an extremal that is a minimum of J, but in this chapter, the solutions are considered to be optimal if any perturbations only increase the value of J. The following subsection presents an GRG method to solve the optimality conditions to determine optimal DOC trajectories.

5.3.2 Numerical Solution Via GRG

The DOC optimality conditions (5.22)-(5.24) consist of a coupled set of parabolic PDEs. Because analytical solutions to these PDEs are not available, this chapter presents a GRG approach for reducing the computation required by the numerical

solution of the DOC optimality conditions. The approach exploits the causality of the macroscopic dynamic equation (5.6) to represent $\hat{J}$ solely as a function of $\mathbf{u}$. Then an extremum of the DOC problem (5.1)-(5.6) can be found by determining the parameters of the control laws (5.10) that satisfy the optimality conditions.

GRG methods improve iteratively upon the approximation of the optimal control law and of the macroscopic state and co-state (or Lagrangian), by holding the other fixed during each update. During every iteration of the GRG algorithm, the latest approximation of $\mathbf{u}^* = \mathbf{c}^*[\mathbf{x}(t), t]$, in parameterized form (5.10) is used to solve macroscopic state and adjoint PDEs, (5.22) and (5.23), to obtain an approximation for $\wp^*$ and λ^*. Subsequently, holding the approximations of $\wp^*$ and λ^* fixed, the approximation for $\mathbf{u}^*$ is updated so as to minimize (5.4), and satisfy the third and final optimality condition. This process is repeated until the norm of the gradient is below a user-defined tolerance or any update to $\mathbf{u}^*$ causes an increase in J.

The GRG method falls under a larger class of optimization techniques referred to as *Nested Analysis and Design* (NAND). In NAND approaches, the gradient is obtained at each iteration of the optimization by eliminating the state and co-state variables by solving the PDEs using a numerical algorithm, and only the control is considered [10]. Alternatively, a *Simultaneous Analysis and Design* (SAND), or full space, optimization strategy could be used in which the optimization over the state, co-state, and control are preformed simultaneously. However, it has been shown that SAND methods are often very ill conditioned, where the individual PDEs in the NAND techniques are typically better conditioned [11].

An analytical representation of the gradient of the cost function, J, with respect to the controls, $\mathbf{u}$, can be found, thereby circumventing the need for finite difference to approximate the gradient, greatly reducing the computational requirements. The gradient of J is calculated as follows. Let $\tilde{\wp}$ and $\tilde{\lambda}$ satisfy (5.22) and (5.23),

Algorithm 1 GRG Optimality Solver

```
initialize α_{j,k}(t)
while ||g|| >TOL do
    φ̃ ← solve macroscopic state PDE (u)
    λ̃ ← solve adjoint PDE (φ̃, u)
    for all ℓ do
        g_ℓ ← compute gradient (φ̃, λ̃, u)
    end for
    for all j, k do
        α_{j,k} ← update α_{j,k} (J, g)
    end for
end while
```

respectively, for a given $\mathbf{u}$. Then the gradient is given by

$$\nabla_u J = \nabla_u \hat{J}\Big|_{\tilde{\wp},\tilde{\lambda}} = \int_{\mathcal{X}} \frac{\partial \phi}{\partial \wp} \nabla_u \wp \; \delta \mathbf{u}\Big|_{\tilde{\wp},t_f} d\mathbf{x} + \tag{5.25}$$

$$\int_t \int_{\mathcal{X}} \left\{ \frac{\partial \mathscr{L}}{\partial \mathbf{u}} \delta \mathbf{u} + \frac{\partial \mathscr{L}}{\partial \wp} \nabla_u \wp \; \delta \mathbf{u} + \nabla_u \lambda \; \delta \mathbf{u} \left[\frac{\partial \wp}{\partial t} + \nabla \cdot (\wp \mathbf{f}) \right] + \right.$$

$$\lambda \left[\frac{\partial}{\partial t} (\nabla_u \wp \; \delta \mathbf{u}) + \nabla \cdot \left(\nabla_u \wp \mathbf{f} \delta \mathbf{u} + \wp \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \delta \mathbf{u} \right) \right]$$

$$\left. - \nu \nabla^2 \nabla_u \wp \delta \mathbf{u} \right\}_{\tilde{\wp},\tilde{\lambda}} d\mathbf{x} dt$$

Performing integration by parts and recalling that $\tilde{\wp}$ and $\tilde{\lambda}$ were defined to satisfy (5.22) and (5.23), equation (5.25) becomes

$$\nabla_u J = \int_t \int_{\mathcal{X}} \left[\frac{\partial \mathscr{L}}{\partial \mathbf{u}} - \nabla \tilde{\lambda} \cdot \left(\tilde{\wp} \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right) \right] \delta \mathbf{u} d\mathbf{x} dt. \tag{5.26}$$

Let the time be discretized into Q equally spaced points, $t_q = t_0 + q\Delta t$, where $q = 0, ..., Q$, and $\Delta t = (t_f - t_0)/Q$. Then from (5.26) it follows that

$$\left. \frac{\partial J}{\partial \alpha_{j,k}} \right|_{t=t_q} \approx \Delta t \int_{\mathcal{X}} \left[\frac{\partial \mathscr{L}}{\partial u_j} - \nabla \tilde{\lambda} \cdot \left(\tilde{\wp} \frac{\partial \mathbf{f}}{\partial u_j} \right) \right]_{t=t_q} \phi_k d\mathbf{x}. \tag{5.27}$$

The previous equation gives the gradient of the cost function with respect to the parameters that determine the control $\mathbf{u}$. Using this expression of the gradient, $\mathbf{u}$

can be updated using one of many gradient-based optimization schemes, such as Sequential Quadratic Programming (SQP). The algorithm for solving the optimality conditions is then given in Algorithm 1. The next section demonstrates the use of Algorithm 1 to find the optimal control law in a multi-agent path planning problem.

5.4 Multi-agent Trajectory Optimization

The GRG method presented in the previous sections is demonstrated here on a multi-agent trajectory optimization problem, that obeys the problem formulation in Section 5.2. Consider a system of N cooperative agents with microscopic dynamics given by a single integrator model for a point robot that was modified from the model proposed in [114],

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} + \sigma \mathbf{I}_2 \begin{bmatrix} \eta_x \\ \eta_y \end{bmatrix} \tag{5.28}$$

where $\mathbf{q} = [x\ y]^T$ denotes the configuration vector of the i^{th} agent, and x and y are the xy-coordinates. The microscopic control vector of the i^{th} agent is $\mathbf{u} = [v_x\ v_y]^T$, where v_x and v_y are linear velocities in the x and y directions, respectively. The disturbance vector is $\mathbf{w} = [\eta_x\ \eta_y]^T$, where η_x and η_y are independent random variables with values given by standard Gaussian processes, σ is a constant, and $\mathbf{I}_2$ is the identity matrix. The agents operate in a workspace $\mathcal{I} = [L,\ 0] \times [0,\ L] \subset \mathbb{R}^2$ over a time interval $(t_0, t_f]$. The system restriction operator is a time-varying PDF of the agent states, $\wp : \mathcal{X} \times \mathbb{R} \to \mathbb{R}$, where $\wp(\mathbf{q}, t)$ provides the probability that the i^{th} agent has the configuration $\mathbf{q}$ at time t. Then $\wp$ describes the density of the agents in the state space $\mathcal{X}$.

The agents have the goal of traveling to a time-invariant target distribution, $p(\mathbf{x})$, that is known *a priori*, while minimizing the energy consumed through control. The objective function to be minimized can be written in terms of $\wp$, and is given by the

integral cost function

$$J(\mathbf{u}) = \int_{\mathcal{I}} \phi(\wp)\Big|_{t_f} d\mathbf{x} + \int_{\mathcal{I}}\int_{t_0}^{t_f} \mathscr{L}(\mathbf{u}) dt d\mathbf{x} = \tag{5.29}$$

$$\int_{\mathcal{I}} w_{\wp}(g-\wp)^2\Big|_{t_f} d\mathbf{x} + \int_{\mathcal{I}}\int_{t_0}^{t_f} e^{\frac{w_u}{2}(v_x^2+v_y^2)} dt d\mathbf{x}$$

where $w_\wp$ and w_u are user-defined constant weights.

The optimal agent PDF $\wp^*$ and control $\mathbf{u}^*$ can be computed as follows. The control, $\mathbf{u}$, is approximated by the Fourier sine series

$$\mathbf{u}(\mathbf{x},t) = \sum_{n=1}^{a}\sum_{m=1}^{b} \sin[n\pi(x_1+1)/2]\sin[m\pi(x_2+1)/2]\alpha_{mnj}(t). \tag{5.30}$$

This form ensures that $\mathbf{u} = 0$ on the boundary, forcing $\mathbf{f}\cdot\hat{\mathbf{n}} = 0$, which simplifies the boundary condition in (5.23) to $\nabla\lambda\cdot\hat{\mathbf{n}} = 0$. With this parameterized representation of the control, the gradient equation (5.27) is given by

$$\frac{\partial J}{\partial \alpha_{qpj}} \approx \Delta t \int_{\mathcal{I}} \Big[w_u u_j e^{\frac{w_u}{2}(v_{xi}^2+v_{yi}^2)} - \wp\frac{\partial\lambda}{\partial x_j}\Big]\times \tag{5.31}$$

$$\sin[p\pi(x+1)/2]\sin[q\pi(y+1)/2] d\mathbf{x}.$$

where u_j and x_j denote the j^{th} component of $\mathbf{u}$ and $\mathbf{x}$, respectively.

The numerical scheme used to solve (5.22) and (5.23) consists of a modified Galerkin method. A Galerkin type method was chosen for its non-dissipative property [50, 93]. In this modified approach, the solution is approximated by the linear combination of a Fourier basis and Gaussian radial basis functions (RBF), which are used to enforce the boundary conditions at each point of the integration.

An initial guess of $\alpha_{qpj} = 0$ was used to define the control for the first iteration of the optimization (Algorithm 1). Then the state and adjoint problems, (5.22) and (5.23), were solved. The control parameters, $\alpha_{qpj}(t_n)$, were then updated using gradient descent.

The agents' feedback control laws can be obtained from a set-point regulation method that uses the optimal agent PDF, $\wp^*$, and open-loop control, $\mathbf{u}^*$, that are found by solving the optimality conditions (5.22)-(5.24), as desired set-points [103]. The closed-loop control of each agent is computed independently, such that the control value at time t of the i^{th} agent, $\mathbf{u}(t)$, is determined to minimize the deviation between the observed agent distribution, denoted as $\hat{\wp}(t)$, and the optimal distribution $\wp^*(t)$, and the deviation between $\mathbf{u}(t)$ and the optimal open-loop control $\mathbf{u}^*(t)$ [83],

$$\begin{aligned} \mathbf{u}^*(t) &= \min_{\mathbf{u}(t)} \int_t^{t+\delta t} \frac{1}{2} \left\{ [\wp^*(\mathbf{x},t) - \hat{\wp}(\mathbf{x},t)]^2 \right. \\ &+ \left. \|\mathbf{u}^*(t) - \mathbf{u}(t)\|^2 \right\} dt \end{aligned} \tag{5.32}$$

where $\| \cdot \|$ is the Euclidean norm, and δt is a user-defined time increment. The observed agent distribution, $\hat{\wp}$, is calculated from the states of all agents using kernel density estimation with a standard Gaussian kernel [98]. The optimal feedback control $\mathbf{u}^*$ is updated at each timestep by minimizing (5.32) using one of several available quadratic programming algorithms [83]. In this chapter, δt is chosen to be small for simplicity, such that $\delta t << t_f - t_0$.

5.4.1 Numerical Simulations

The GRG method presented in Section 5.3 is illustrated here through a numerical example where the optimal agent trajectories are calculated for a system of $N = 250$ agents with microscopic dynamics governed by the single integrator model (5.28) with $\sigma = 0.01$. The agents exist in a workspace $\mathcal{I} = [0,\ L] \times [0,\ L]$, $L = 16$ km, over a time interval $(t_0,\ t_f]$, where $t_0 = 0$ and $t_f = 16$ hr. The agents have a given initial distribution g_0 shown in Figure 5.1(a), and the initial microscopic states and sampled from g_0. The system's objective is to minimize the integral cost (5.29) with $w_\wp = 100$, $w_u = 6$, by travelling to a known target agent distribution p, illustrated

in Figure 5.1(b), while minimizing the energy consumed by control. The solution to the trajectory optimization problem is found using the GRG approach, and the optimal agent distribution and microscopic states are plotted at four instants in time in Figure 5.2. It is seen from the results that the optimal agent distribution $\wp^*$ reaches the target distribution p.

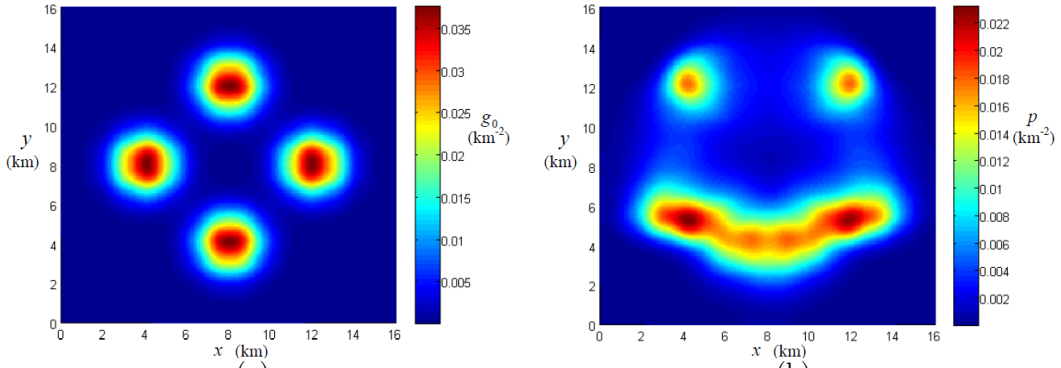


FIGURE 5.1: Initial agent distribution, g_0 (a). Target agent distribution, p (b).

The agents' control input is given by the feedback control law (5.32) and calculated using MATLAB's quadratic program solver *quadprog*, where $\delta t = 20$ s. Then the microscopic states are updated by integrating the microscopic dynamic equations (5.28). The optimal microscopic state trajectories of $s = 50$ randomly-chosen agents are plotted in Figure 5.3(a), and the optimal microscopic control trajectories of $r = 3$ randomly-chosen agents are shown in Figure 5.3(b).

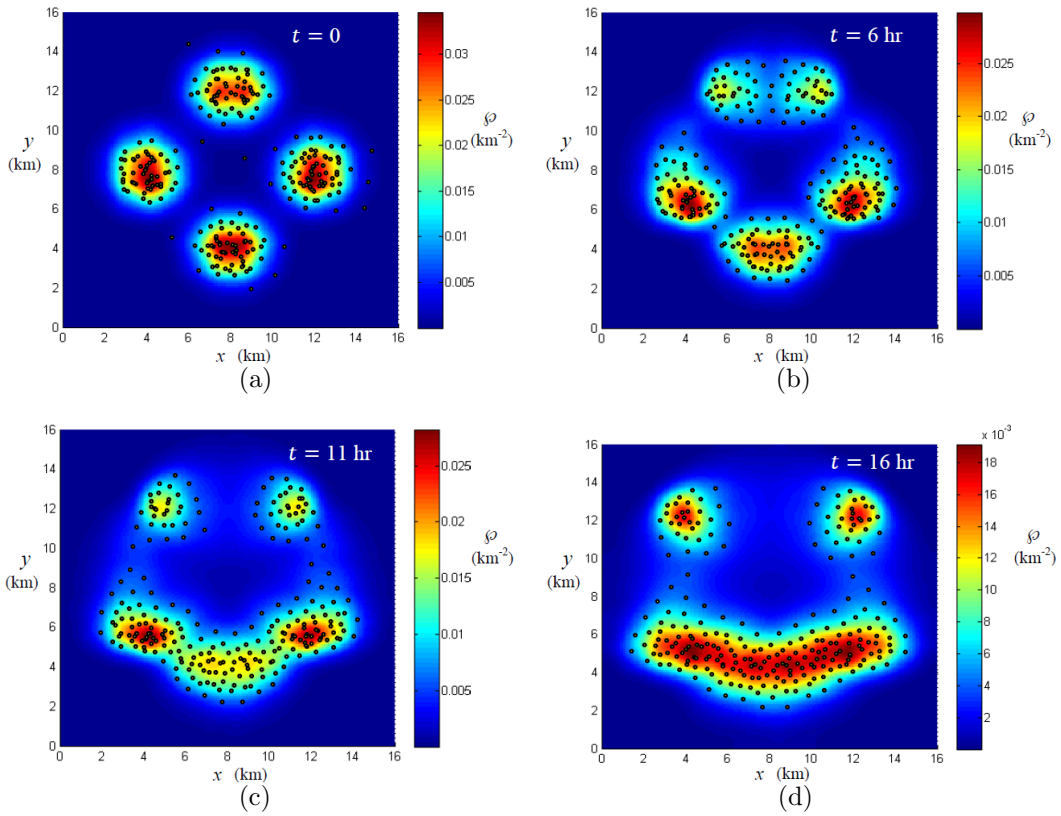


FIGURE 5.2: Optimal evolution of agent distribution and microscopic states (yellow circles) for a system of $N = 250$ agents at four instants in time.

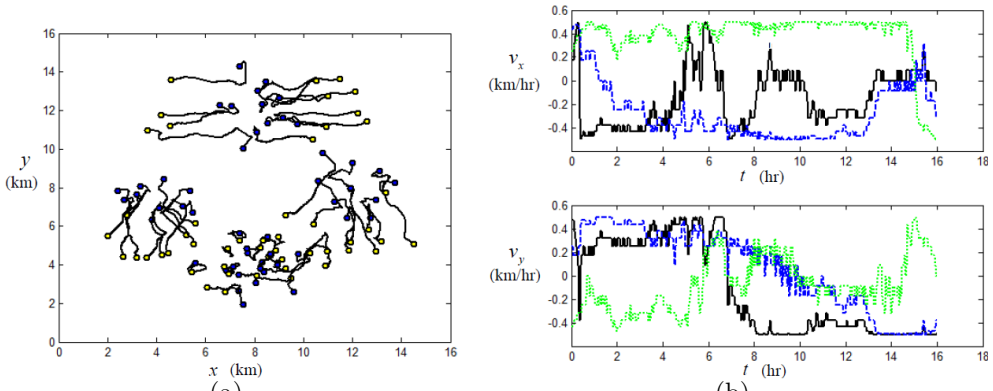


FIGURE 5.3: Optimal microscopic state trajectories of $s = 50$ randomly-chosen agents traveling from their initial states (blue circles) to final states (yellow circles) (a). Optimal microscopic control trajectories of $r = 3$ randomly-chosen agents.

6

Optimal Root Profiles in Water-Limited Ecosystems

A plant's growth, reproduction, and survival all depend on the plant's ability to absorb soil moisture through its root system [60, 66]. As root distributions are determined by the survival strategy of the plant, optimization concepts have been used to identify root distributions based on ecohydrological facets of the soil-plant-atmosphere system [67]. This chapter focuses on water-limited ecosystems and identify optimal vertical root profiles that maximize transpiration in order to explore how a shift in the temporal structure of rainfall might affect competition between different rooting strategies, as well as how herbaceous plants would need to adjust their root profile to remain optimal in its access to water.

Knowledge of the active root layer is essential for the study of water and nutrient dynamics as needed in atmospheric science, hydrology, ecology, and geochemistry (e.g. Bhattachan et al. [9]). There are several factors that influence root depths and distributions. For example, Schenk and Jackson found a positive correlation between rooting depths and annual potential evapotranspiration (PET), mean annual

precipitation (MAP), and length of the warm season [90]. In particular, Schenk and Jackson were able to use MAP to explain 62% of the observed variance in median rooting depths for herbaceous plants in water-limited ecosystems [91].

It has also been shown that root distributions [47] and absolute root depths [16] vary by vegetation type. As mentioned above, Schenk and Jackson found a positive correlation between MAP and median root distributions for herbaceous plants [91, 90], however, Bhattachan et al. showed that this correlation may not apply to woody root distributions [9].

Two optimization variables that are frequently used to infer characteristics of root distributions are carbon gain and transpiration. Kleidon and Heimann [52] estimated optimal root depths by maximizing the carbon gain to the vegetation within a global Terrestrial Biosphere Model. Schwinning and Ehleringer explored potential trade-offs in water uptake and carbon cost by developing a simple model of plant water transport and carbon gain in a two-layered soil environment [92]. Similarly, Guswa provided a cost-benefit analysis of root structures [38, 39], where the optimal root depth was balanced by the carbon cost of forming the root structure.

Transpiration optimality was first used to predict root characteristics in [82], where Protopapas and Bras identified root profiles which maximize transpiration in a maize crop in Flevoland (Netherlands). They used a transient soil moisture model, primarily driven by initial conditions, as they assumed no precipitation during the simulations. A similar approach was taken in [22], where Collins and Bras simulated soil moisture content that was driven at the surface by an average daily rainfall. van Wijk also used an average daily rainfall in his analysis to show that observed patterns of rooting characteristics of herbaceous plant species could be explained using the concept of hydrological optimality for arid climates [110]. In [57] Laio et al. used a steady-state analytical model of soil moisture to link vertical root distributions to climate and soil properties. Specifically, they investigated the dependency of

the optimal average root depth on average daily rainfall and potential transpiration (PT) for various soil types. Sivandran and Bras also used optimal transpiration to investigate the relationship between soil type and optimal root profiles in [101]. In their analysis they used a stochastic rainfall model to generate rainfall data for a single climate.

Recent analysis of the climate system [21] has suggested that increased greenhouse forcing can lead to mechanistic changes in precipitation frequency and intensity. In fact, there is a likely tendency towards decreased frequency and increasing intensity in the tropics. This raises the question of how root profiles may need to adapt to simultaneous changes in storm depth and frequency. This work further explores the use of hydraulic optimization as a means of predicting vertical characteristics of root profiles for herbaceous plants. However, unlike [82, 22, 101, 110, 57], this study focuses on the effects that storm structure may have on optimal root profiles in order to better understand how plants may need to adapt to a changing climate. Richards' equation is used to model soil moisture with seasonal potential evapotranspiration (PET) and leaf area index (LAI). The soil moisture is driven at the surface by seasonal stochastic rainfall to observe changes in optimal root profiles occurring from perturbations in storm frequency and intensity. As this work focuses on water-limited ecosystems, the carbon cost of deeper roots or the role of other nutrients are not considered.

For each climate type and trial root profile, Richards equation is solved using a new constrained integration (CINT) partial differential equation (PDE) solution method. The CINT method is similar to pseudo-spectral solution methods; in this method traditional Galerkin methods are combined with a modified constrained backpropogation (CPROP) [31, 26] algorithm in which radial basis functions (RBFs) are used to enforce the boundary conditions.

This chapter is organized as follows: Section 6.1 gives a mathematical description

of the governing equations used in the model. Section 6.2 describes the climate and soil types used in the mathematical simulations. The results of the simulations are given in Section 6.3, followed by a brief discussion in Section 6.4.

6.1 Model Description

A transient state of soil moisture is used that is driven at the surface by rain and evaporation. In this model, roots compete for moisture with evaporative effects near the soil surface and with gravity drainage at lower depths. It is assumed that interception is small, and that rainfall rates in excess of the maximum infiltration rate are lost as runoff. Additionally, only vertical water fluxes are considered and it is assume that the root zone does not interact with the water table below. The variables and parameters used in the model are summarized in Table 6.1.

Groundwater flow within the vadose zone was modeled using Richards' equation [19]:

$$\frac{\partial \theta}{\partial t}(z,t) = \frac{\partial}{\partial z}\left\{K[\psi(z,t)]\left[\frac{\partial \psi}{\partial z}(z,t)+1\right]\right\} - S(z,t). \tag{6.1}$$

The soil moisture, $\theta(z,t)$, and matric potential, $\psi(z,t)$, states are related here by van Genuchten's formula [107],

$$\theta(\psi) = \theta_r + (\theta_s - \theta_r)S_e(\psi), \tag{6.2}$$

where

$$S_e(\psi) = [1+(\alpha\psi)^n]^{-m}. \tag{6.3}$$

The hydraulic conductivity, K, was approximated using Mualem's formula [72],

$$K(\psi) = k_s S_e^{\ell}(\psi)\left[1-(1-S_e^{1/m}(\psi))^m\right]^2. \tag{6.4}$$

The values used for the soil parameters pertaining to the soil types used are shown in Table 6.2.

Table 6.1: Description of model variables and parameters.

Variable	Description
z	Vertical distance (positive up) $[cm]$
t	Time $[d]$
ψ	Matric Potential $[cm]$
θ	Volumetric water content $[cm^3/cm^3]$
K	Hydraulic conductivity $[cm/d]$
S	Root sink term $[d^{-1}]$
$\mathcal{R}$	Precipitation rate $[cm/d]$
E	Evaporation $[cm/d]$
T	Total transpiration
ρ	Root density $[1/cm]$
γ	Root efficiency term
PET	Potential evapotranspiration $[cm/d]$
PE	Potential evaporation $[cm/d]$
PT	Potential transpiration $[cm/d]$
LAI	Leaf area index $[m^2/m^2]$
θ_r	Residual water content $[cm^3/cm^3]$
θ_s	Saturated water content $[cm^3/cm^3]$
α	Fitting parameter $[cm^{-1}]$
n	Fitting parameter (dimensionless)
m	Fitting parameter (dimensionless)
k_s	Saturated soil conductivity $[cm/d]$
ℓ	Fitting parameter (dimensionless)

The sink term, $S(z,t)$, is the rate at which moisture is extracted from the soil by the root system at time t and depth z, and is described by

$$S = \gamma(\psi)\rho(z)PT, \tag{6.5}$$

where $\gamma(\psi)$, shown in Fig. 6.1, is the root efficiency term given by van Genuchten [109]

$$\gamma(\psi) = \frac{1}{1 + (\psi/\psi_{50})^p}. \tag{6.6}$$

In the above equation, ψ_{50} is the soil-water pressure head at which the extraction rate is reduced by 50%, and in the work presented in this chapter was given the value of 50 cm, similar to values reported in [112]. The parameter p is commonly assumed

to have a value of 3 (dimensionless) [108].

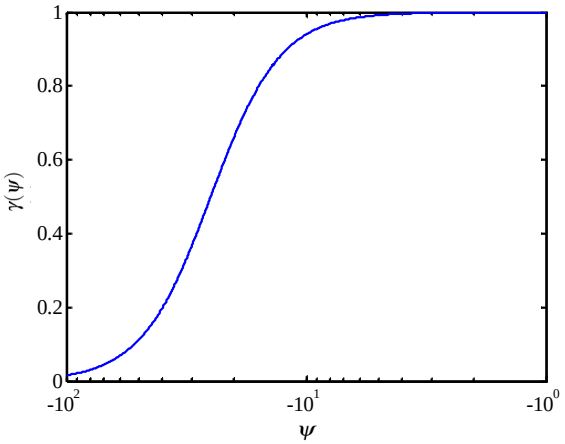


FIGURE 6.1: Plot of the root efficiency term, $\gamma(\psi)$.

PET is partitioned into potential evaporation, PE, and potential transpiration, PT [85],

$$PE = PET(e^{-\sigma LAI}) \tag{6.7}$$

$$PT = PET(1 - e^{-\sigma LAI}), \tag{6.8}$$

where $\sigma = .4$ [22]. Seasonal LAI, with values similar to those reported in [81, 49], was represented by,

$$LAI(t) = 1 + \tanh(3\cos(2\pi t/365)). \tag{6.9}$$

A plot of (6.9) is shown in Fig. 6.2. It is also assumed seasonal PET, with a peak daily total of 5 mm/d during the wet/growing season and 1.25 mm/d during the dry season [104]. This is represented by

$$PET(t) = .5\pi[3/8(1 + \cos(2\pi t/365)) + .25]\times \max\{\sin(2\pi t), 0\}. \tag{6.10}$$

An analytical function of shape for the root density is assumed, given by

$$\rho(z) = \frac{c(z/D_{50})^{c-1}}{D_{50}\left[1 + (z/D_{50})^c\right]^2}, \tag{6.11}$$

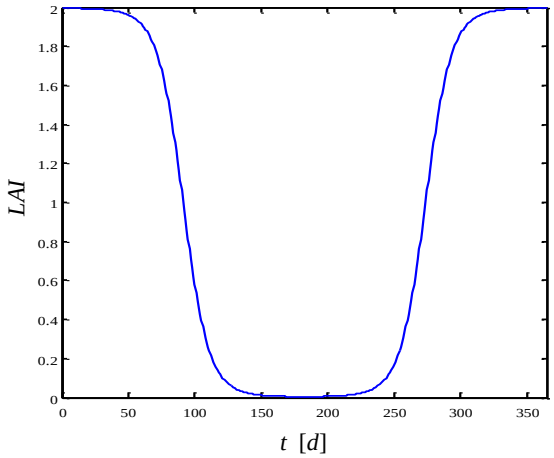


FIGURE 6.2: One season of LAI as a function of t [81].

where c is defined as

$$c = \frac{2.94}{\ln(D_{50}/D_{95})}. \tag{6.12}$$

D_{50} and D_{95} are parameters that determine the shape of $\rho(z)$. D_{50} determines the depth at which the bulk of the root density is located; specifically, it is the depth above which 50% of the root is located. The parameter D_{95} is associated with absolute rooting depths, and is the depth above which 95% of the root is located. This density function is the derivative of the cumulative density function proposed by Schenk and Jackson [90] and has been widely used to describe root distributions of herbaceous plants [110, 22]. An example density is shown in Fig. 6.3 for $D_{50} = -100$ $[cm]$ and $D_{95} = -200$ $[cm]$.

This work explores how, for fixed MAP, varying storm frequency and intensity affect the optimal root profile. The focus is on relative changes in root distribution rather than absolute rooting depth, and so, it was assumed that $D_{95} = -200$ $[cm]$ in order to simplify the analysis. The parameter D_{50} was allowed to vary to maximize

the total transpiration,

$$T = \int_0^{10\times 365} \int_{-400}^{0} S(z,t)\, dz\, dt. \tag{6.13}$$

A 10 year period was chosen as the time frame for the simulations. In initial trials it was found that this was a sufficient amount of time for clear patterns to emerge in the results.

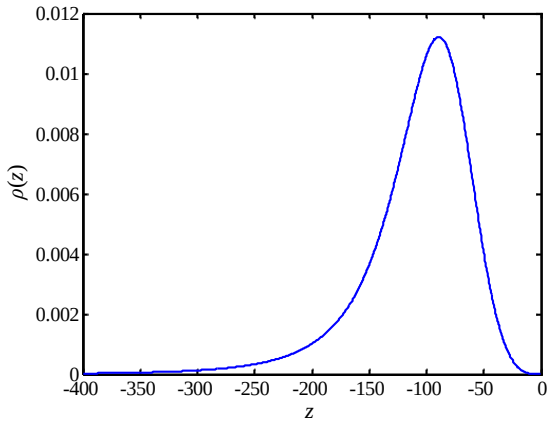


FIGURE 6.3: Example root density profile, $\rho(z)$, with $D_{50} = -100$ and $D_{95} = -200$.

The boundary condition at the surface ($z = 0$) is given as a specified flux (Neumann condition) [19] using Darcy's law

$$\mathcal{R}(t) - E[\psi(0,t)] = K(\psi)\left(\frac{\partial \psi}{\partial z} + 1\right)\Big|_{z=0}, \tag{6.14}$$

where $r(t)$ is the rainfall rate and $E[\psi(0,t)]$ is the evaporation rate given by

$$E = \beta[S_e(\psi(0,t))]PE. \tag{6.15}$$

In the above equation, PE is the potential evaporation specified by (6.8) and $\beta[S_e(\psi(0,t))]$ is a function that describes the effect of water stress on soil evaporation. The water stress function ranges from 0 to 1, and in this study is assumed to have the form

$$\beta[S_e(\psi(0,t))] = \frac{1}{2}\left[1 + \tanh\left(c_2 S_e^{c_3} - \frac{c_1}{S_e}\right)\right]_{z=0}, \tag{6.16}$$

where c_1, c_2, and c_3 are positive constants. A plot of $\beta[S_e(\psi(0,t))]$ is shown in Fig. 6.4. Note that this function's output is similar to piecewise functions commonly used (see [22, 110]), however is differentiable everywhere.

This work seeks to examine modest perturbations in key features of a climate's precipitation structure. A two parameter stochastic rainfall generator was used to obtain the rainfall time series used in these simulations. With this generator, precipitation is represented as a Poisson process with mean storm frequency, λ^* $[d^{-1}]$, and mean storm depth α^* $[mm]$ [38, 86, 28, 23].

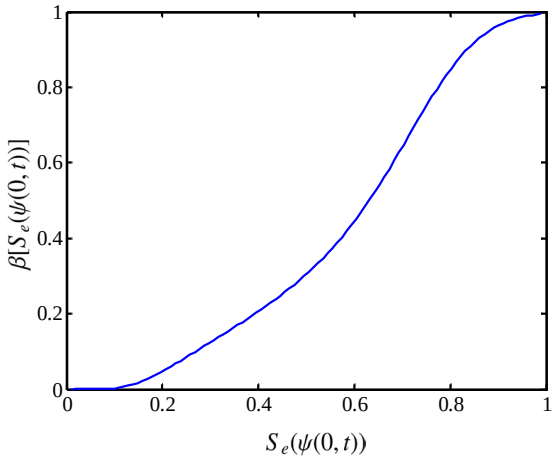


FIGURE 6.4: Water stress function $\beta[S_e(\psi(0,t))]$ used to approximate the rate of evaporation.

The boundary condition at the bottom of the soil profile, taken to be at $z = -400$ cm, is given by drainage under gravity (e.g. [5]) and was implemented by enforcing

$$\left.\frac{\partial \psi}{\partial z}\right|_{z=-400} = 0. \tag{6.17}$$

Richards' equation (6.1) was solved using the CINT method [54]. The CINT method is similar to pseudo-spectral solution methods; in this method traditional Galerkin methods are combined with a modified CPROP algorithm [31, 26], in which RBFs are used to enforce the boundary conditions. The CINT method was chosen

because it has been shown to be faster and more accurate than the finite element (FE) method for several types of PDEs [54].

To ease the numerical implementation in the application of the CINT method to Richards' equation, particularly near saturation, the substitution $\tilde{\psi}(z,t) = \log[-\psi(z,t)]$ was made. This substitution changes the PDE (6.1) to

$$\frac{\partial\tilde{\psi}}{\partial t}\frac{\partial\theta}{\partial\psi} = \frac{\partial\tilde{\psi}}{\partial z}\frac{\partial K}{\partial\psi}\left[1 - \exp(\tilde{\psi})\frac{\partial\tilde{\psi}}{\partial z}\right] + K\left[\left(\frac{\partial\tilde{\psi}}{\partial z}\right)^2 + \frac{\partial^2\tilde{\psi}}{\partial z^2}\right] + \exp(-\tilde{\psi})S. \tag{6.18}$$

Note that the boundary condition at the bottom of the soil profile given in (6.17) does not change, as

$$\frac{\partial\psi}{\partial z} = -\exp(\tilde{\psi})\frac{\partial\tilde{\psi}}{\partial z} \tag{6.19}$$

implies that the boundary condition for the re-scaled hydraulic pressure head is

$$\frac{\partial\tilde{\psi}}{\partial z}\Big|_{z=-400} = 0. \tag{6.20}$$

The depth, z, was rescaled and shifted to $\tilde{z}$,

$$\tilde{z} = z/200 + 1, \tag{6.21}$$

so that $\tilde{z} \in [-1, 1]$.

As done in Galerkin and pseudo-spectral methods, the re-scaled hydraulic pressure head, $\tilde{\psi}(\tilde{z},t)$, is approximated by the linear combination of basis functions,

$$\tilde{\psi}(\tilde{z},t) \approx \sum_{j=1}^{J}\phi_j(\tilde{z})\alpha_j(t). \tag{6.22}$$

The boundary condition (6.20) is enforced by setting

$$\alpha_1(t) = -\left(\frac{\partial\phi_1(\tilde{z})}{\partial\tilde{z}}\right)^{-1}\sum_{j=2}^{J}\frac{\partial\phi_j(\tilde{z})}{\partial\tilde{z}}\alpha_j(t)\Big|_{\tilde{z}=-1} \tag{6.23}$$

In the above equation, $\phi_1(z)$ is the Gaussian function

$$\phi_1(z) = \exp[-100(z+1.1)^2]. \tag{6.24}$$

For PDE problems that do not contain traveling waves, commonly used basis functions such as Fourier or Chebyshev polynomials can be used for $\phi_j(\tilde{z})$, however, as traveling waves appear in solutions to (6.1), piecewise cubic polynomials were found to be more effective. Letting the domain, $[-1,1]$ be partitioned into $J-1$ equally spaced subregions $[\tilde{z}_j, \tilde{z}_{j+1}]$, the functions $\phi_j(\tilde{z})$ for $j = 2, ..., J$ are piecewise cubic polynomials given in by,

$$\phi_j = \frac{1}{\Delta z^3}\begin{cases} (\tilde{z}-\tilde{z}_{j-2})^3, \text{ if } z \in [\tilde{z}_{j-2}, \tilde{z}_{j-1}] \\ \Delta z^3 + 3\Delta z^2(\tilde{z}-\tilde{z}_{j-1}) + 3\Delta z(\tilde{z}-\tilde{z}_{j-1})^2 - 3(\tilde{z}-\tilde{z}_{j-1})^3, \text{ if } \tilde{z} \in [\tilde{z}_{j-1}, \tilde{z}_j] \\ \Delta z^3 + 3\Delta z^2(z_{j+1}-z) + 3\Delta z(\tilde{z}_{j+1}-\tilde{z})^2 - 3(\tilde{z}_{j-1}-\tilde{z})^3, \text{ if } \tilde{z} \in [\tilde{z}_j, \tilde{z}_{j+1}] \\ (\tilde{z}_{j+2}-\tilde{z})^3, \text{ if } \tilde{z} \in [\tilde{z}_{j+1}, \tilde{z}_{j+2}] \\ 0, \text{ otherwise} \end{cases} \tag{6.25}$$

where $\Delta z = \tilde{z}_{j+1} - \tilde{z}_j$.

To solve Richards' equation, the right hand side of (6.23) is substituted into (6.22), giving an approximate solution that satisfies the boundary condition at the bottom of the soil profile. This approximate solution is then substituted into (6.18), and evaluated at collocation points within the domain $[-1,1]$, resulting in a system of ordinary differential equations (ODEs). This system of ODEs is then integrated to obtain an approximate solution to (6.18). In this work, the integration algorithm of choice was Matlab's *ODE15s* [94]. At each integration time step, t_j, the approximate solution was re-constructed at the collocation points by evaluating (6.22). The approximate solution was then adjusted at the soil surface so as to satisfy the boundary condition (6.15), using fixed point iterations to deal with the nonlinearity in the condition.

6.2 Experimental Setup

This work focuses on the effects of simultaneous changes in storm frequency and intensity on optimal root distributions. This section specifies climate and soil properties that were used in the model description given in the previous section. The Kalahari is used as an interesting example, but the work is intended to illuminate the general case of changing precipitation structure in water-limited ecosystems.

6.2.1 Affect of Precipitation on Root Depth

To better understand the effects that storm type has on optimal root profiles, rainfall data were generated for eleven year periods, with varying frequency and intensity of storms. Mean storm frequency was varied from 0.1 to 0.4 d^{-1}, and mean depth varied from 8 to 12 mm, similar to values reported by Porporato et al. [79] for a transect of the Kalahari. Storm frequency, λ^*, was allowed to vary seasonally,

$$F_m = \left[.5(1 + \tanh(\cos(2\pi m/12))\right]^4, \tag{6.26}$$

$$\lambda_m = \lambda^* F_m / \overline{\mathbf{F}}, \tag{6.27}$$

$$\lambda_m^* = \min\{\lambda_m, .99\}. \tag{6.28}$$

where m indicates the month of year, and λ_m^* is the average storm frequency for month m. The average depth was also allowed to vary seasonally,

$$\alpha_m^* = \alpha^* + 3.5 \tanh(\cos(2\pi m/12 + \pi)). \tag{6.29}$$

From the eleven year period, the first year was used for 'spin-up' to minimize the effects of the initial conditions. Simulations were then run over the remaining 10 years for analysis. The soil type used in these numerical experiments was sandy loam, and the parameters associated with this soil are given in the following subsection.

6.2.2 Sensitivity to Underlying Soil Type

This work also explores the sensitivity of the results to underlying soil type by obtaining the optimal root profiles for two different soil types. These simulations focus on the wet end of the Kalahari, near Manu Zambia, with mean storm depth $\alpha^* = 10.1$ mm and mean arrival time $\lambda^* = .38$ d^{-1} [79]. The average monthly totals for this site are plotted in Fig. 6.5, and are in close agreement with the monthly averages reported in [79].

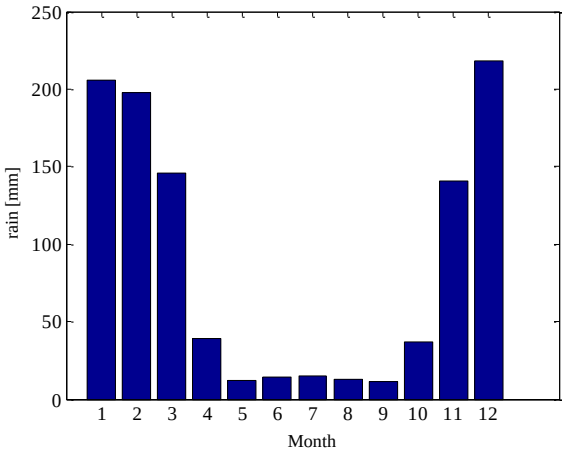


FIGURE 6.5: Average monthly rainfalls generated for $\alpha^* = 10.1$ mm and $\lambda^* = .38$ d^{-1}.

The two soil types considered in this work were sandy loam and sandy clay loam. The parameters associated with these soil types are found in Table 6.2.

Table 6.2: Soil specific parameters used in (6.2) and (6.4).

	Sandy Loam	Sandy Clay Loam
θ_s	.41	.39
θ_r	.065	.1
α	-.075	-.059
n	1.89	1.49
m	$1-n$	$1-n$
ℓ	.5	.5
k_s	106.1	31.44

6.3 Simulations and Results

6.3.1 Variable Storm Type

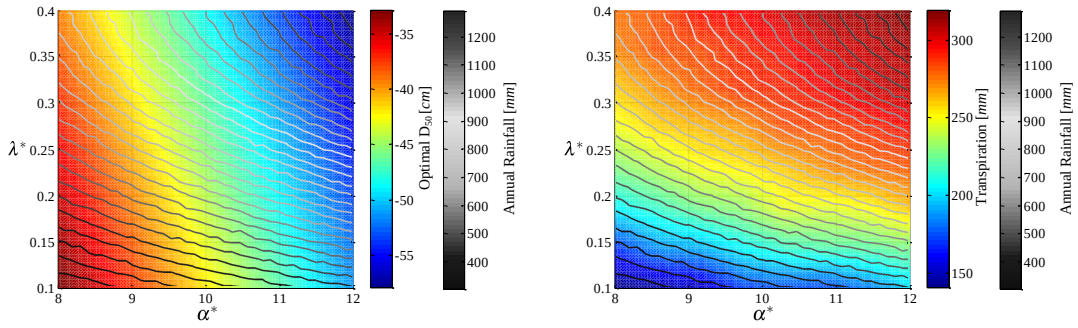
The results of the simulations across storm types are given in Figs. 6.6(a)-6.7(b). The contour lines in these figures show constant values of MAP. In Fig. 6.6(a) the optimal D_{50} is shown over the values of λ^* and α^* that were explored. These results indicate that for low MAP the optimal profile is distributed closer to the surface, and becomes more deeply distributed as MAP increases, as shown in [82, 22, 101, 110, 57]. This suggest that, for areas with low MAP, plants compete with evaporation for water. However, as MAP increases, the roots must compete with evaporation and drainage for moisture and, hence, a deeper distribution becomes advantageous.

What is new in this study is that one can observe variations in optimal root structures for differing storm types with a fixed MAP. These findings provide a view of below ground implications for predicted changes in α^* and λ^* [21]. Traveling along a contour line of constant MAP in Fig. 6.6(a), one observes that as frequency increases and depth decreases, shallower roots are advantageous. As frequency decreases and intensity increases, the optimal root profile is more deeply distributed.

The average yearly transpiration, drainage, and evaporation recorded in the simulations with the optimal root profiles (Fig.6.6(a)) are shown in Figs. 6.6(b)-6.7(b). Change in storage is was not included for brevity, as it was small (on the order of the observed error). These figures show that transpiration, drainage, and evaporation remain nearly constant along lines of constant MAP, provided the vegetation is able to adapt to the new storm frequency and intensity.

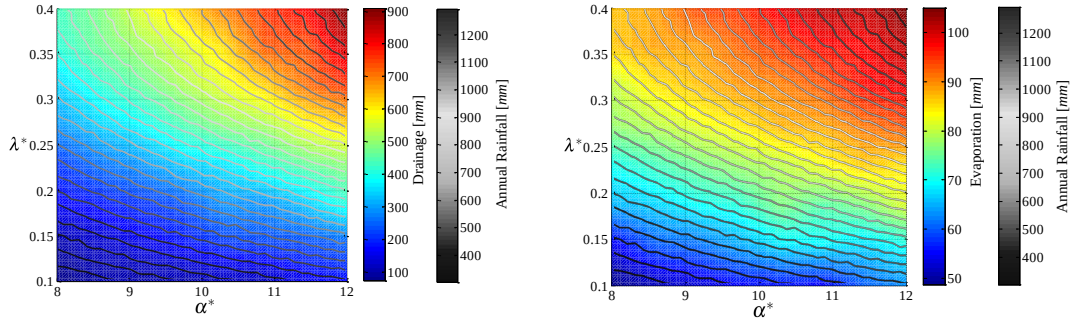
6.3.2 Variable Soil Type

The results of the simulations over varying soil types are shown in Figs. 6.8(a)-6.9(a). Figure 6.8(a) shows recorded transpiration as a function of D_{50} for the two



(a) (b)

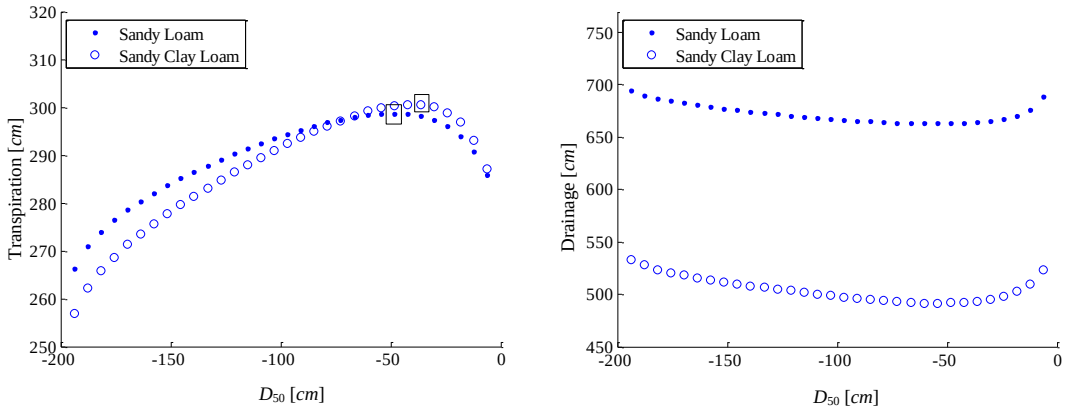
FIGURE 6.6: Optimal value of D_{50} versus mean storm depth, α^*, and mean arrival time, λ^* (a). Mean annual transpiration as a function of mean storm depth, α^*, and mean arrival time, λ^* (b).



(a) (b)

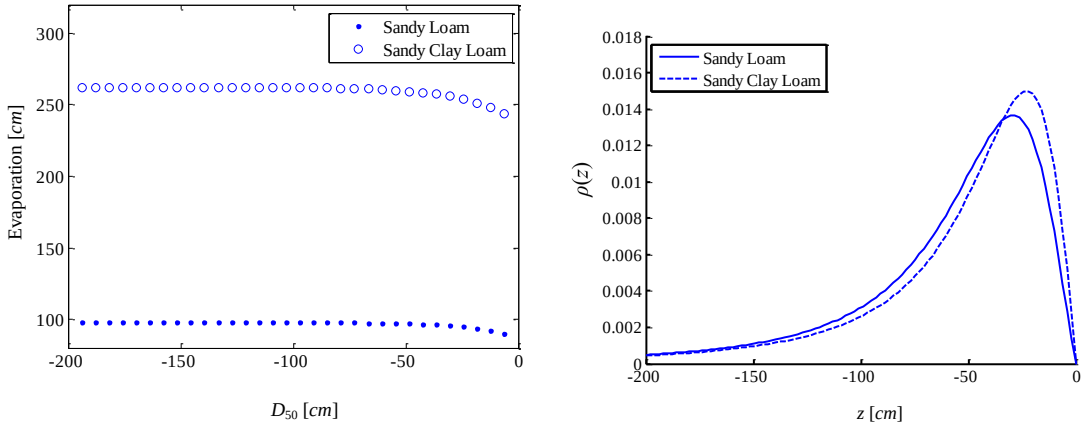
FIGURE 6.7: Mean annual drainage as a function of mean storm depth, α^*, and mean arrival time, λ^* (a). Mean annual evaporation as a function of mean storm depth, α^*, and mean arrival time, λ^* (b).

soil types used in this study, where the optimal D_{50} has been indicated. Figure 6.8(b) shows the observed drainage, and Fig. 6.9(a) shows the observed evaporation. The corresponding optimal profiles for each soil are shown in Fig. 6.9(b). These results show, as others have observed (see [101, 22, 110, 57]), that for soils with high conductivity, the optimal profile is more deeply distributed. This characteristic also agrees with what has been observed in nature [91, 47].



(a) (b)

FIGURE 6.8: Transpiration versus D_{50} for each soil type (a). Drainage versus D_{50} for each soil type.



(a) (b)

FIGURE 6.9: Evaporation versus D_{50} for each soil type(a). Optimal root profiles for each soil type (b).

6.4 Discussion

The results in the previous section show that for soils with a sandier constitution and a higher hydraulic conductivity, the optimal root profiles are more deeply distributed, as observed in nature [91, 47]. These results are in harmony with previous work [82, 22, 101, 110, 57]. The water balances shown in Figs. 6.8(a)-6.9(a) suggest that for sandier soils evaporative effects are not as significant as the effects of gravity,

and, thus, a deeper distribution is optimal.

The results given for optimal profiles (Fig. 6.6(a)) as a function of storm intensity and frequency show the dependence of the optimal profile on mean annual precipitation. These results empirically agree with what was reported in [91], that root systems tend to be deeper and narrower in cold and wet climates and more shallowly distributed in hot, dry climates.

The results shown in Fig. 6.6(a) also show that as storms become less frequent and more intense, the optimal profile has a deeper distribution. It is interesting to note that the optimal profile returns approximately the same water balance for each storm type, suggesting that in order to maintain current rates of transpiration, plants may need to adjust rooting strategies in response to predicted climatic changes [21].

These results raise interesting questions for future studies of nutrient dynamics and implications for below ground carbon allocation.

7

Conclusions and Recommendations

ANNs have been used in a number of applications to provide functional representations of PDE solutions that are amenable to mathematical analysis, and to more efficient processing by data assimilation and estimation algorithms. In many of these applications, however, the underlying dynamic process may be subject to change as a result of a non-stationary environment. Thus, while the PDE may capture the dynamic process on short time scales, the process, and thus the PDE, both are subject to change over long time scales. Typically, the presence of I/BCs equality constraints makes the optimization problem more difficult to solve, because it reduces the set of feasible solutions. However, a well known result from constrained optimization theory is that if the equality constraints satisfy the implicit function theorem, they can be written in explicit form, and used to reduce the dimensionality of the optimization problem through the method of direct elimination. In this case, the optimization problem is simplified, and the equality constraints are satisfied exactly at every iteration of the optimization algorithm.

It was recently shown that the method of direct elimination can be used to train ANNs in the presence of equality constraints through CPROP. This thesis shows that

CPROP offers a general and natural paradigm for solving PDEs in non-stationary environments because the ANN can be adapted to minimize the error defined by the differential operator, while satisfying initial and boundary conditions through direct elimination. Furthermore, direct elimination can be applied by means of an adjoined gradient or Jacobian that simplifies the computation of error derivatives across the hidden layer subject to the equality constraints. The effectiveness of the CPROP solution method is demonstrated through several examples of linear and nonlinear elliptic and parabolic PDEs, subject to initial and boundary conditions. The method also is applicable to irregular domains, and to other classes of PDEs, including hyperbolic equations. For both elliptic and parabolic equations, CPROP brings about a significant reduction in the number of iterations required for solving the PDE adaptively, and is characterized by a computational complexity and a solution accuracy that compare favorably to existing methods of solution.

Additionally, the CPROP based approach was extended to the CINT method for solving IBVPs. It was shown how the CINT method combines classical Galerkin methods with CPROP in order to constrain the ANN to approximately satisfy the boundary condition at each stage of integration. The advantage of the CINT method is that it is readily applicable to PDEs in irregular domains and requires no special modification for domains with complex geometries. Furthermore, the CINT method provides a semi-analytical solution that is infinitely differentiable. The CINT method was implemented on three IBVPs with different domains and boundary conditions. These problems were chosen because they have simple analytical solutions with which comparisons where made. For the hyperbolic problems, the CINT method outperformed Matlab's FE method in terms of speed and accuracy. In the first IBVP the CINT method reduced the computational time by 85% and the error by 70%. In the second problem, the CINT method reduced the computational time by 94% and the error by 88%. Finally, it was shown that for these problems, the CINT method

exhibits the exponential rate of convergence seen in classical Galerkin methods.

The CINT method was used in solving the optimality conditions that arose in a DOC problem, giving the optimal state and control trajectories for a multiscale dynamical system comprised of many interacting dynamical systems, or agents. A GRG approach is presented to compute the optimal agent state and control trajectories for the DOC problem formulation with stochastic agent dynamics. This expands the capabilities of the DOC approach, which was previously only formulated for systems with deterministic agent dynamics. A new set of optimality conditions are derived for this case and are then solved using an indirect optimization method with GRG to obtain a functional representation of the optimal macroscopic state and microscopic open-loop control. A microscopic feedback control law is obtained using a set-point regulation method. The optimality conditions and the GRG approach are verified through a numerical simulation that determines the optimal state and control trajectories of a large system of agents with dynamics governed by a single integrator point robot model.

The CINT method was also used to show that hydraulic optimality can be used to identify root distributions with characteristics that are in agreement with root profiles observed in nature for water-limited ecosystems. In particular, optimal root profiles were identified for sandy clay loam and sandy loam soil types for climates typical of the Kalahari.

The optimal profiles were identified for a spectrum of storm types, with mean depths ranging from 8 to 12 mm and mean frequencies ranging from .1 to .4 d^{-1}. It was shown that the optimal profile depends, not only on mean annual precipitation, but also on the storm type. These results suggest that as forcing from greenhouse gases result in shifting storm structure [21], plants in water limited ecosystems will be required to adapt their rooting strategies in order to maintain optimality and water balances.

The work done in this thesis has yielded several interesting questions for future research. One possible area to explore is developing the CPROP PDE solver to be applicable to IBVPs with Neumann or Robin boundary conditions. Future research could also focus on developing the method to be capable of solving hyperbolic type equations. There are also interesting aspects of the CINT method to explore. For example, future research could include identifying the optimal shape parameters and number of RBF used, and how these relate to a given PDE and type of STM transfer functions.

Appendix A

Partial Derivative of ANN Solution for Parabolic Problems

The second partial derivative of the ansatz (3.31) to the parabolic IBVP is given by,

$$\frac{\partial^2 \hat{u}(\mathbf{x})}{\partial \mathbf{x}_{(j)} \partial \mathbf{x}_{(k)}} = \frac{\partial^2 \tilde{f}(\mathbf{x})}{\partial \mathbf{x}_{(j)} \partial \mathbf{x}_{(k)}} + \frac{\partial^2 q(\mathbf{x})}{\partial \mathbf{x}_{(j)} \partial \mathbf{x}_{(k)}} [\mathbf{\Phi}(\mathbf{x}^T \mathbf{W}_L^T +$$

$$\mathbf{b}_L^T) \mathbf{v}_L^T + \mathbf{\Phi}(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{b}_S^T) \mathbf{v}_S^T] + \frac{\partial q(\mathbf{x})}{\partial \mathbf{x}_{(j)}} [\mathbf{\Phi}^1(\mathbf{x}^T \mathbf{W}_L^T$$

$$+ \mathbf{b}_L^T) \boldsymbol{\omega}_{L_k} \mathbf{v}_L^T + \mathbf{\Phi}^1(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{x}_S^T) \boldsymbol{\omega}_{S_k} \mathbf{V}_S^T] + \frac{\partial q(\mathbf{x})}{\partial \mathbf{x}_{(k)}} \times$$

$$[\mathbf{\Phi}^1(\mathbf{x}^T \mathbf{W}_L^T + \mathbf{b}_L^T) \boldsymbol{\omega}_{L_j} \mathbf{v}_L^T + \mathbf{\Phi}^1(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{b}_S^T) \times$$

$$\boldsymbol{\omega}_{S_j} \mathbf{v}_S^T] + q(\mathbf{x}) [\mathbf{\Phi}^2(\mathbf{x}^T \mathbf{W}_L^T + \mathbf{b}_L^T \boldsymbol{\omega}_{L_j} \boldsymbol{\omega}_{L_k} \mathbf{V}_L^T$$

$$+ \mathbf{\Phi}^2(\mathbf{x}^T \mathbf{W}_S^T + \mathbf{b}_S^T) \boldsymbol{\omega}_{S_j} \boldsymbol{\omega}_{S_k} \mathbf{v}_S^T] \tag{A.1}$$

Appendix B

Finite Difference Stencil

The FDM stencil used to solve the Boussinesq equation (4.10) is given by

$$
\begin{aligned}
S^m_{n_{i,j}} \frac{u^{m+1}_{i,j} - u^m_{i,j}}{\Delta \mathbf{p}_{(3)}} = & \left(\frac{\partial K_n}{\partial \mathbf{p}_{(1)}} \right)^m_{i,j} u^m_{i,j} \frac{u^m_{i+1,j} - u^m_{i-1,j}}{2\Delta \mathbf{p}_{(1)}} + \left(\frac{\partial K_n}{\partial \mathbf{p}_{(2)}} \right)^m_{i,j} u^m_{i,j} \frac{u^m_{i,j+1} - u^m_{i,j+1}}{2\Delta \mathbf{p}_{(2)}} \\
& + K^m_{n_{i,j}} \left[\left(\frac{u^m_{i+1,j} - u^m_{i-1,j}}{2\Delta \mathbf{p}_{(1)}} \right)^2 + \left(\frac{u^m_{i,j+1} - u^m_{i,j-1}}{2\Delta \mathbf{p}_{(2)}} \right)^2 \right] \\
& + K^m_{n_{i,j}} u^m_{i,j} \left(\frac{u^m_{i+1,j} - 2u^m_{i,j} + u^m_{i-1,j}}{\Delta \mathbf{p}^2_{(1)}} + \frac{u^m_{i,j+1} - 2u^m_{i,j} + u^m_{i,j-1}}{\Delta \mathbf{p}^2_{(2)}} \right).
\end{aligned} \tag{B.1}
$$

Bibliography

[1] *MATLAB Neural Network Toolbox, User's Guide*. The MathWorks, 2005.

[2] *MATLAB Partial Differential Equation Toolbox, User's Guide*. The MathWorks, 2005.

[3] Najib N. Abboud and Peter M. Pinsky. Finite element dispersion analysis for the three-dimensional second-order scalar wave equation. *International Journal for Numerical Methods in Engineering*, 35(6):1183–1218, 1992.

[4] M. Ainsworth, P. Monk, and W. Muniz. Dispersive and dissipative properties of discontinuous galerkin finite element methods for the second-order wave equation. *Journal of Scientific Computing*, 27(1-3):5–40, 2006.

[5] J.D. Albertson and G. Kiely. On the structure of soil moisture time series in the context of land surface models. *Journal of Hydrology*, 243(12):101 – 119, 2001.

[6] J Attali and G Pages. Approximations of functions by a multilayer perceptron: a new approach. *Neural Networks*, 10(6), 1997.

[7] T. Balch and R. C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14, 1998.

[8] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, Belmont, MA, 1996.

[9] Abinash Bhattachan, Mokganedi Tatlhego, Kebonye Dintwe, Frances O'Donnell, Kelly K. Caylor, Gregory S. Okin, Danielle O. Perrot, Susan Ringrose, and Paolo D'Odorico. Evaluating ecohydrological theories of woody root distribution in the kalahari. *PLoS ONE*, 7(3):e33996, 03 2012.

[10] L.T. Biegler. *Large-Scale Pde-Constrained Optimization*. Lecture Notes in Computational Science and Engineering, 30. Springer Verlag, 2003.

[11] G. Biros and O. Ghattas. Parallel lagrange–newton–krylov–schur methods for pde-constrained optimization. part i: The krylov–schur solver. *SIAM Journal on Scientific Computing*, 27(2):687–713, 2005.

[12] G. H. Bower. *The psychology of learning and motivation: advances in research and theory*. Academic Press, 1989.

[13] J. P. Boyd. *Chebyshev and Fourier Spectral Methods, II Ed.* Dover, New York, NY, 2001.

[14] JohnP. Boyd. Large-degree asymptotics and exponential asymptotics for fourier, chebyshev and hermite coefficients and fourier transforms. *Journal of Engineering Mathematics*, 63(2-4):355–399, 2009.

[15] Alfonso Bueno-Orovio, Victor Perez-Garcia, and Flavio H. Fenton. Spectral methods for partial differential equations in irregular domains: The spectral smoothed boundary method. *SIAM Journal on Scientific Computing*, 28(3):886–15, 2006.

[16] J. Canadell, R. B. Jackson, J. R. Ehleringer, H. A. Mooney, O. E. Sala, and E.-D. Schulze. Maximum rooting depth of vegetation types at the global scale. *Oecologia*, 108(4):583–595, 1996.

[17] C. Canuto. *Spectral methods in fluid dynamics*. Springer series in computational physics. Springer-Verlag, 1988.

[18] C.G. Canuto, Y. Hussaini, and A. Quarteroni. *Spectral Methods: Fundamentals in Single Domains*. Scientific Computation. Deutsches MAB-Nationalkomitee beim Bundesministerium für Umwelt, Naturschutz und Reaktorsicherheit, 2006.

[19] Andrés Cesanelli and Luis Guarracino. Estimation of actual evapotranspiration by numerical modeling of water flow in the unsaturated zone: a case study in buenos aires, argentina. *Hydrogeology Journal*, 17(2):299–306, 2009.

[20] T Cheng, F L Lewis, and M Abu-Khalaf. Fixed-final-time-constrained optimal control of nonlinear systems using neural network hjb approach. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, 18(6), 2007.

[21] Chia Chou, Chao-An Chen, Pei-Hua Tan, and Kuan Ting Chen. Mechanisms for global warming impacts on precipitation frequency and intensity. *Journal of Climate*, 25(9):3291–3306, 2012.

[22] D B G Collins and R L Bras. Plant rooting strategies in water-limited ecosystems. *WATER RESOURCES RESEARCH*, 43, 2007.

[23] Edoardo Daly, A. Christopher Oishi, Amilcare Porporato, and Gabriel G. Katul. A stochastic model for daily subsurface {CO2} concentration and related soil respiration. *Advances in Water Resources*, 31(7):987 – 994, 2008.

[24] J. W. Delleur. *Groundwater Engineering*. CRC Press, Boca Raton, FL, 2007.

[25] J. P. Desai, J. Ostrowski, and V. Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17:905908, 2001.

[26] G. Di Muro and S. Ferrari. A constrained-optimization approach to training neural networks for smooth function approximation and system identification. In *Proc. International Joint Conference on Neural Networks*. Hong Kong, 2008.

[27] M W M G Dissanayake and N Phan-Thien. Neural-network-based aproximations for solving partial differential equations. *Communications in numerical methods in engineering*, 10:195–201, 1994.

[28] Laio F., Porporato A., Ridolfi L., and Rodriguez-Iturbe I. Plants in water-controlled ecosystems: active role in hydrologic processes and response to water stress - ii. probabilistic soil moisture dynamics. *Advances in Water Resources*, 24(7), 2001.

[29] L. Pollini F. Giulietti and M. Innocenti. Autonomous formation flight. *IEEE Control Systems Magazine*, 20:3444–3457, 2000.

[30] S. Ferrari. Multiobjective algebraic synthesis of neural control systems by implicit model following. *Neural Networks, IEEE Transactions on*, 20(3):406 –419, march 2009.

[31] S. Ferrari and M. Jensenius. A constrained optimization approach to preserving prior knowledge during incremental training. *IEEE Trans. On Neural Networks*, 19(6):996–1009, 2008.

[32] S. Ferrari and R.F. Stengel. Smooth function approximation using neural networks. *IEEE Trans. On Neural Networks*, 16(1):24–38, 2005.

[33] Roger Fletcher and Sven Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91(2):239–269, 2002.

[34] G. Foderaro and S. Ferrari. Necessary conditions for optimality for a distributed optimal control problem. *Proc. IEEE Conference on Decision and Control, Atlanta, Georgia*, 2010.

[35] V. Gazi and K.M. Passino. Stability analysis of social foraging swarms. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(1):539 –557, feb. 2004.

[36] D. Gottlieb and S.A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1977.

[37] B.Y. Guo. *Spectral methods and their applications*. World Scientific Publishing Company, Incorporated, 1998.

[38] A. J. Guswa. The influence of climate on root depth: A carbon cost-benefit analysis. *Water Resources Research*, 44(2), 2008.

[39] A. J. Guswa. Effect of plant uptake strategy on the wateroptimal root depth. *Water Resources Research*, 46(9), 2010.

[40] K Hornik, M Stinchcombe, and H White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3, 1990.

[41] L.E. Howle. A comparison of the reduced galerkin and pseudo-spectral methods for simulation of steady rayleigh-bnard convection. *International Journal of Heat and Mass Transfer*, 39(12):2401 – 2407, 1996.

[42] T. J. R. Hughes. *The finite element method*. New Jersey: Prentice Hall, 1987.

[43] T.J.R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, Incorporated, 2012.

[44] M. Hüsken, C. Goerick, and A. Vogel. Fast adaptation of the solution of differential equations to changing constraints. In *Proceedingg of the Second International ICSC Symposium on Neural Computation*. 2000.

[45] B. Igelnik and Yoh-Han Pao. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. *Neural Networks, IEEE Transactions on*, 6(6):1320 –1329, 1995.

[46] I.D. Iliev, E.K. Khristov, and K.P. Kirčev. *Spectral methods in soliton equations*. Monographs and Surveys in Pure and Applied Mathematics Series. Longman Scientific & Technical, 1994.

[47] R. B. Jackson, J. Canadell, J. R. Ehleringer, H. A. Mooney, O. E. Sala, and E. D. Schulze. A global analysis of root distributions for terrestrial biomes. *Oecologia*, 108:389–411, 1996.

[48] Elham Javidmanesh, Zahra Afsharnezhad, and Sohrab Effati. Bifurcation analysis of a cellular nonlinear network model via neural network approach. *Neural Computing and Applications*, pages 1–6, 2013.

[49] William M Jolly and Steven W Running. Effects of precipitation and soil water potential on drought deciduous phenology in the kalahari. *Global Change Biology*, 10(3):303–308, 2004.

[50] M.R Kaazempur-Mofrad and C.R Ethier. An efficient characteristic galerkin scheme for the advection equation in 3-d. *Computer Methods in Applied Mechanics and Engineering*, 191(46):5345 – 5363, 2002.

[51] I.G. Kevrekidis, C.W. Gear, J.M. Hyman, P.G. Kevrekidis, O. Runborg, , and C. Theodoropoulos. Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis. *Communications in Mathematical Sciences*, 1(4):715–762, 2003.

[52] Axel Kleidon and Martin Heimann. A method of determining rooting depth from a terrestrial biosphere model and its impacts on the global water and carbon cycle. *Global Change Biology*, 4(3):275–286, 1998.

[53] D.A. Kopriva. *Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers.* Scientific computation. Springer, 2009.

[54] K.Rudd and S. Ferrari. A constrained integration (cint) approach to solve partial differential equations with artificial neural networks. *Neurocomputing, In Review*, 2013.

[55] I.E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Trans. On Neural Networks*, 9(5):987–1000, 1998.

[56] I.E. Lagaris, A. Likas, and D. Papageorgio. Neural-network methods for boundary value problems whith irregular boundaries. *IEEE Trans. On Neural Networks*, 11(5):1041–1049, 2000.

[57] F Laio, P D'Odorico, and L Ridolfi. An analytical model to relate the vertical root distribution to climate and soil properties. *GEOPHYSICAL RESEARCH LETTERS*, 33, 2006.

[58] S.M. LaValle and S.A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *Robotics and Automation, IEEE Transactions on*, 14(6):912 –925, dec 1998.

[59] Y. LeCun. A theoretical framework for backpropagation. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 21–28, CMU, Pittsburgh, Pa, 1988. Morgan Kaufmann.

[60] C. A. Lee and W. K. Lauenroth. Spatial distributions of grass and shrub root systems in the shortgrass steppe. *American Midland Naturalist*, 132(1):pp. 117–123, 1994.

[61] N. E. Leonard and E. Fiorelli. Virtual leaders, artificial potentials and coordinated control of groups. In *Proceedings of the Conference on Decision and Control*, page 29682973, 2001.

[62] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168, 1944.

[63] X Li. Simultaneous approximations of multivariate functions and their derivatives by neural networks with one hidden layer. *Neurocomputing*, (12):327–343, 1996.

[64] S W Liu, J H Huang, J C Sung, and C C Lee. Detection of cracks using neural networks and computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 191(2526):2831 – 2845, 2002.

[65] Wenxin Liu, J. Sarangapani, G.K. Venayagamoorthy, D.C. Wunsch, and D.A. Cartes. Neural network based decentralized excitation control of large scale power systems. In *Neural Networks, 2006. IJCNN '06. International Joint Conference on*, pages 1975 –1981, 2006.

[66] J Lynch. Root architecture and plant productivity. *Plant Physiol.*, 109(1):7–13, 1995.

[67] A. Mäkelä, T.J. Givnish, F. Berninger, T.N. Buckley, G.D. Farquhar, and P. Hari. Challenges and opportunities of the optimality approach in plant ecology. *Silva Fennica*, 36(3):605–614, 2002.

[68] X. Mao. *Stochastic Differential Equations and Applications*. Horwood Publishing, 1997.

[69] D. W. Marquardt. An algorithm for least-squares estimation of non-linear parameters. *Journal of the Society of Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[70] K. S. McFall and J. R. Mahan. Artificial neural network method for solution of boundary value problems with exact satisfaction of arbitrary boundary conditions. *IEEE Trans. on Neural Networks*, 20(8), 2009.

[71] A.J. Jr. Meade and A.A. Fernandez. Solution of nonlinear ordinary differential equations by feedforward neural networks. *Mathematical and Computer Modelling*, 20(9):19 – 44, 1994.

[72] Yechezkel Mualem. A new model for predicting the hydraulic conductivity of unsaturated porous media. *Water Resources Research*, 12(3):513–522, 1976.

[73] Gianluca Di Muro and Silvia Ferrari. A constrained-optimization approach to training neural networks for smooth function approximation and system identification. In *IJCNN'08*, pages 2353–2359, 2008.

[74] P O'Neil. *Advanced Engineering Mathematics*. Cengage Learning, Stamford, CT, 2012.

[75] Steven A Orszag. Spectral methods for problems in complex geometries. *Journal of Computational Physics*, 37(1):70 – 92, 1980.

[76] S. Peng and Z. Wu. Fully coupled forward-backward stochastic differential equations and applications to optimal control. *SIAM Journal on Control and Optimization*, 37(3):825843, 1999.

[77] R. Peyret. *Spectral Methods for Incompressible Viscous Flow*. Number v. 148 in Applied Mathematical Sciences. Springer, 2002.

[78] A.D. Pierce. *Acoustics: an introduction to its physical principles and applications*. Acoustical Soc. of America (American Inst. of Physics), 1989.

[79] Amilcare Porporato, Francesco Laio, Luca Ridolfi, Kelly K. Caylor, and Ignacio Rodriguez-Iturbe. Soil moisture and plant stress dynamics along the kalahari precipitation gradient. *Journal of Geophysical Research: Atmospheres*, 108(D3), 2003.

[80] W. Press, S. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 1986.

[81] JL Privette, RB Myneni, Y Knyazikhin, M Mukelabai, G Roberts, Y Tian, Y Wang, and SG Leblanc. Early spatial and temporal validation of modis lai product in the southern africa kalahari. *Remote Sensing of Environment*, 83(1):232–243, 2002.

[82] Angelos L Protopapas and Rafael L Bras. A model for water uptake and development of root systems. *Soil Science*, 144(5), 1987.

[83] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003.

[84] J. H. Reif and H. Wang. Social potential fields: A distributed behavioral control for autonomous robots. *Robotics and Autonomous Systems*, 27(3):171–194, 1999.

[85] Joe T Ritchie. Model for predicting evaporation from a row crop with incomplete cover. *Water Resources Research*, 8(5):1204–1213, 1972.

[86] I. Rodriguez-Iturbe, A. Porporato, L. Ridolfi, V. Isham, and D. R. Coxi. Probabilistic modelling of water balance at a point: the role of climate, soil and vegetation. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 455(1990):3789–3805, 1999.

[87] L Rogers and F Dowla. Optimization of groundwater remediation using artificial neural networks with parallel solute transport modeling. *Water Resources Research*, 30(2):457–481, 1994.

[88] C Saloma. Computational complexity and the observation of physical signals. *J. Appl. Phys.*, 74(9), 1993.

[89] J. Sanchirico and J. Wilen. Optimal spatial management of renewable resources: Matching policy scope to ecosystem scale. *Journal of Environmental Economics and Management*, 50, 2005.

[90] H J Schenk and R B Jackson. The global biogeography of roots. *Ecological Monographs*, 72:311–328, 2002.

[91] H J Schenk and R B Jackson. Rooting depths, lateral root spreads and belowground/above-ground allometries of plants in water-limited ecosystems. *Jornal of Ecology*, 90:480–494, 2002.

[92] Susanne Schwinning and James R Ehleringer. Water use trade-offs and optimal adaptations to pulse-driven arid ecosystems. *Journal of Ecology*, 89(3):464–480, 2001.

[93] T.K. Sengupta, S.B. Talla, and S.C. Pradhan. Galerkin finite element methods for wave problems. *Sadhana*, 30, 2005.

[94] L. Shampine and M. Reichelt. The matlab ode suite. *SIAM Journal on Scientific Computing*, 18(1):1–22, 1997.

[95] R Shekari Beidokhti and a Malek. Solving initial-boundary value problems for systems of partial differential equations using neural networks and optimization techniques. *Journal of the Franklin Institute*, 346, 2009.

[96] R.K. Shevgaonkar. *Electromagnetic Waves*. Electrical & electronic engineering series. McGraw-Hill Education (India) Pvt Limited, 2005.

[97] Y. Shirvany, M. Hayati, and R. Moradian. Multilayer perceptron neural networks with novel unsupervised training method for numerical solution of the partial differential equaitons. *Applied Soft Computing*, 9(1), 2009.

[98] J. S. Simonoff. *Smoothing Methods in Statistics*. Springer, 1996.

[99] A. Singh and J. P. Hespanha. Moment closure techniques for stochastic models in population biology. *IEEE Proceedings of the American Control Conference*, pages 4730–4735, 2006.

[100] A. Singh and J. P. Hespanha. A derivative matching approach to moment closure for the stochastic logistic model. *Bulletin of Mathematical Biology*, 69(6):1909–1925, 2007.

[101] Gajan Sivandran and Rafael L. Bras. Identifying the optimal spatially and temporally invariant root distribution for a semiarid environment. *Water Resources Research*, 48(12), 2012.

[102] G. D. Smith. *Numerical solution of partial differential equations: Finite difference methods*. Oxford: ClarendonPress, 1978.

[103] R. F. Stengel. *Optimal Control and Estimation*. Dover Publications, Inc., 1986.

[104] D. Stephenson, E.M. Shemang, and T.R. Chaoka. *Water Resources of Arid Areas: Proceedings of the International Conference on Water Resources of Arid and Semi-Arid Regions of Africa, Garborone, Botswana, 3-6 August 2004*. Taylor & Francis, 2004.

[105] W. Sun and Y.X. Yuan. *Optimization Theory and Methods: Nonlinear Programming*. Springer Optimization and Its Applications Series. Springer Science+Business Media, LLC, 2006.

[106] S. Thrun, M. Bennewitz, and W. Burgard. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. *Robotics and Autonomous Systems*, 41(2):89–99, 2002.

[107] van Genuchten. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil Science Society of America Journal*, 44(5):892–898, 1980.

[108] M Th Van Genuchten and SK Gupta. A reassessment of the crop tolerance response function. *JOURNAL-INDIAN SOCIETY OF SOIL SCIENCE*, 41:730–730, 1993.

[109] M.T. Van Genuchten and U.S. Salinity Laboratory. *A Numerical Model for Water and Solute Movement in and Below the Root Zone*. United States Department of Agriculture Agricultural Research Service U.S. Salinity Laboratory, 1987.

[110] Mark T. van Wijk. Understanding plant rooting patterns in semi-arid systems: an integrated model analysis of climate, soil type and plant biomass. *Global Ecology and Biogeography*, 20(2):331–342, 2011.

[111] R. Verfürth. A posteriori error estimates for finite element discretizations of the heat equation. *CALCOLO*, 40(3):195–212, 2003.

[112] J A Vrugt, M T van Wijk, J W Hopmans, and J Simunek. One-,two-, and three-dimensional root water uptake functions for transient modeling. *WATER RESOURCES RESEARCH*, 37(10):2457–2470, 2001.

[113] Paul Werbos. Backwards differentiation in ad and neural nets: Past links and new opportunities. In Martin Bücker, George Corliss, Uwe Naumann, Paul Hovland, and Boyana Norris, editors, *Automatic Differentiation: Applications, Theory, and Implementations*, volume 50 of *Lecture Notes in Computational Science and Engineering*, pages 15–34. Springer Berlin Heidelberg, 2006.

[114] M.M. Zavlanos, M.B. Egerstedt, and G.J. Pappas. Graph-theoretic connectivity control of mobile robot networks. *Proceedings of the IEEE*, 99(9):1525 –1540, sept. 2011.

Biography

Keith Clive Rudd was born on Dec. 27, 1982 in Salt Lake City, UT (USA). He earned a bachelor of science degree in mathematics from Brigham Young University in 2007. His emphases were in applied mathematics and numerics. During his undergraduate he participated in research of the stability of viscous shocks and received an ORCA grant to support his summer research. He, along with another undergraduate, presented their research at the SIAM conference on dynamical systems in Snowbird, UT.

Keith received a master of science degree in engineering sciences and applied mathematics from Northwestern University in 2008, a master of engineering management degree from Duke University in 2010, and a Ph.D. in mechanical engineering from Duke University in 2013. During his Ph.D. work he has been supported by the NSF as a IGERT WiseNet fellow.